ENERGY-EFFICIENT COMMUNICATION STRATEGIES FOR WIRELESS SENSOR NETWORKS

A Thesis
Presented to
The Academic Faculty

by

Yujie Zhu

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Electrical and Computer Engineering

Georgia Institute of Technology
August 2007

ENERGY-EFFICIENT COMMUNICATION STRATEGIES FOR WIRELESS SENSOR NETWORKS

Approved by:

Dr. Raghupathy Sivakumar, Advisor
Department of ECE
Georgia Institute of Technology

Dr. Ian F. Akyildiz
Department of ECE
Georgia Institute of Technology

Dr. George Riley
Department of ECE
Georgia Institute of Technology

Dr. Biing Hwang Juang
Department of ECE
Georgia Institute of Technology

Dr. Mostafa H. Ammar
Department of CoC
Georgia Institute of Technology

Date Approved: 27th April 2007

To my family.

ACKNOWLEDGEMENTS

I am grateful to the help of many individuals, without whom this dissertation would not have been possible.

First and foremost, I would like to express my sincere gratitude to my advisor, Dr. Raghupathy Sivakumar. He taught me the methodologies and approaches for academic research, and the essential skills of presenting research works. He also gave me freedom in making my own decisions and supported me along the way. During my PhD study years, I have always been amazed and inspired by his strive for excellence, passion for research, and his innovative mind. He is truly my role model.

I am also indebted to my advisor for the opportunity of pursuing my research in the GNAN group, an inspiring and productive research group he created. Thanks to his thoughtful guidance and management, our research group is a great environment for developing skills, exchanging ideas, and eventually generating high-quality research works. I remember the numerous insightful paper discussions, the idea-stimulating brainstorming sessions, and the constructive presentation dry-runs. It is my great honor to be a member of the GNAN group. This is an unique experience that I will always be proud of and benefit from for the rest of my life.

I would like to thank my dissertation committee members: Dr. Ian F. Akyildiz, Dr. George Riley, Dr. Biing Hwang Juang and Dr. Mostafa H. Ammar for serving in my thesis proposal and defense committee. I am grateful for the time they spent in reading my thesis and their valuable advices and comments, which helped me improve the quality of this thesis.

I extend my thanks to my current and past labmates in the GNAN research group. Special thanks to Sandeep, Karthik, Ram and Seung-Jong for their collaborations and suggestions in my thesis research. Thanks to Tae-Young for helping me with computer related technical problems and sharing great deal information with me. Thanks to Hung-Yun and Vaidy for setting up good examples as researchers during my junior years in the GNAN group. Thanks to Sriram for helping me with the administrative issues of the thesis defense when I was not in Atlanta. I also thank Kyu-Han, Aravind, Ashraf, Zhenyun, Yeonsik and Chen-Lin for their feedbacks and offers of help during my thesis research and defense.

Finally I would like to thank my family: my parents, Zhu Puji and Feng Faqing, my sister, Zhu Yongxuan and my niece Long Lingxiao for always being there for me. Your encouragement and advices helped me through the good times and bad times. Your care and support will always accompany me through my adventures.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

Wireless sensor networks (WSNs) are characterized by limited amount of energy supply at sensor nodes. Hence, energy efficiency is an important issue in system design and operation of WSNs.

In this work we focus on solving the energy efficiency problems of *data gathering processes* in WSNs. We first address this problem on a macroscopic level by investigating the efficiency of data gathering trees when data sent by different sensors are *correlated*. Such correlation aware data gathering strategies typically shift the aggregation structure from a default shortest-path tree (SPT) to a steiner minimum tree (SMT) in order to achieve the required efficiency.

We study the energy efficiency of correlation aware data aggregation trees under various sensor network conditions and the tradeoffs involved in using them. Comprehensive simulations results as well as inferences and theoretical analysis of those results are presented in the thesis.

Based on the insights gained through the investigation, we propose a simple, scalable and distributed correlation aware aggregation structure that achieves good energy performance across a wide range of sensor network configurations, and at the same time addresses the practical challenges of establishing a correlation aware data aggregation structure in resource-constrained WSNs.

On a microscopic level, we propose a novel communication strategy called *Communication through Silence* (CtS) to achieve energy-efficient data gathering without significant degradation on overall throughput in WSNs. The proposed scheme primarily uses time, along with a minimal amount of energy to deliver information among sensors. CtS can be used to replace the conventional energy-based transmissions between each pair of sensor nodes during a data gathering process. We analyze in detail the primary energy-throughput

tradeoff inherent in this approach as well as other challenges related to the realization of the proposed communication strategy. Finally, we propose a practical realization of CtS strategy that includes radio technology, MAC layer, and higher layer solutions. Performance evaluation results prove that this solution effectively realizes the CtS strategy in a WSN setting, at the same time achieves considerable energy savings compared to conventional communication strategies.

CHAPTER I

INTRODUCTION

1.1 Objective of the PhD Research

Wireless Sensor Networks (WSNs) have garnered a considerable amount of attention over the last half a decade, primarily due to the unique applications they enable. However, one of their advantages, the ability to be deployed in otherwise inaccessible environments, also lays an important constraint on the operation of such networks - the energy source at sensors. Except for environments where an energy source can be harnessed in a low cost manner, the very survivability of WSNs depends upon how energy efficiently the sensors operate in performing their required functions.

While there have been numerous efforts at developing routing and communication protocols for WSNs [9], the primary concern is the energy efficiency of all these solutions. This consideration potentially affect many aspects of the system design: hardware, physical layer, MAC layer, addressing and routing, topology control, synchronization, naming scheme, security mechanisms, etc. Numerous solutions [21,32,37,44,55,56] have been proposed in this context, with each of them addressing one or more aspects of WSN system design.

In this research work, we focus on the primary operation in any WSN: collection of sensor data from the sensors in the field to the sink for processing (the *data gathering* process), and aim to improve the energy efficiency of such a process. Since data gathering is the most important and frequent operation in a WSN, energy gain obtained through the optimization of this process can help extend the lifetime of sensor networks significantly.

The problem is tackled on two different levels in our research. On the macroscopic level, we consider the problem of data gathering in environments where data from the different sensors are correlated. Such correlation of the data being collected can be leveraged by appropriately fusing the data inside the network to the best extent possible using an energy-efficient aggregation tree rooted at the sink, thereby reducing energy consumption for the gathering process.

In the first part of the research, we study the energy efficiency of correlation aware data aggregation trees under various sensor network conditions and the tradeoffs involved in using them [57]. The following three related questions are specifically investigated in the study: (i) Is there any practical limitation on the achievable improvement in energy efficiency in adopting a correlation aware aggregation structure as opposed to a correlation unaware structure? (ii) Is there a practically maximum useable delay bound that can deliver the maximum achievable improvement? (iii) How does the correlation degree between sensor data affect the optimal aggregation structure? In answering the above questions, we present comprehensive simulation results and draw inferences from the results.

Based on insights obtained from the investigation, we propose a scalable correlation aware data gathering structure called *SCT* (Semantic/Spatial Correlation-aware Tree) to achieve the energy improvement without incurring significant overhead [58]. SCT is a simple, scalable, and distributed data aggregation approach that achieves considerable energy improvement with relatively low overhead under a wide range of sensor application scenarios. SCT addresses the challenges of constructing and maintaining data aggregation structures in resource constrained sensor networks, as well as practical problems such as load balancing, synchronization and fault tolerance.

On the microscopic level, we propose a novel communication paradigm that enables energy-efficient data gathering in WSNs in a per-hop fashion. Compared with traditional communication solutions, the proposed scheme explores a new dimension - *time*, combined with minimal energy to deliver information efficiently. This communication strategy is referred to as *Communication through Silence* (CtS) [63]. We identify a key drawback of such a strategy - *the energy-throughput trade-off*, and discuss challenges related to the realization of the new communication strategy. We also propose a practical realization of CtS strategy including radio technology, MAC layer, and higher layer solutions in this thesis.

1.2 Overview of the Thesis Organization

The rest of this proposal is organized as follows: Chapter 2 provides background and an overview of the macroscopic solution. Chapter 3 presents in detail the study of correlation aware data gathering process in WSNs. Chapter 4 describes the scalable and efficient correlation aware data aggregation solution and explains the rationale behind its design. Chapter 5 explain the background of the microscopic solution, and Chapter 6 introduces the concept of the communication through silence strategy we proposed. In Chapter 7, we present a practical realization of the CtS strategy and the performance evaluation results of the solution. Finally, we conclude the thesis in Chapter 8.

CHAPTER II

BACKGROUND

On the macroscopic level, an important challenge associated with data gathering in WSNs is to reduce the message complexity [1] to minimize the bandwidth usage of the network and the energy consumption of the sensor nodes by taking advantage of *correlations* among sensor data. Here correlation refers to the information redundancy between two sensor messages due to the overlap of sensing activities in the spatial, temporal or semantic domain.

The following two types of correlations are considered in this paper:

- *Spatial Correlation:* This refers to the correlation of the data reported by multiple sensors sensing the same event or phenomenon. For example, consider the query: What is the temperature in the region defined by the rectangle (x1,y1,x2,y2)? Given the typical dense deployments of sensors in WSNs, it is likely that the sensing regions of two different sensors within the rectangular region overlap. Consequently, the data reported by these sensors are spatially correlated. If the two sensors are very close to each other, the data reported by both sensors is practically the same, which implies that the sensors are perfectly correlated.

- *Semantic Correlation:* This refers to the correlation of data reported by multiple sensors due to the semantics of the query. The messages from different sensors report different events or phenomena and hence the content of these messages may not be spatially correlated. However, if the query imposed is not about the specific details of each event, but about certain statistics over the entire event region, it is likely that the messages generated by each source can still be aggregated. For example, consider the query: *Is the total number of cars in the rectangular region (x1,y1,x2,y2) greater than K?* In this case, even if the sensors are reporting data about *different* cars,

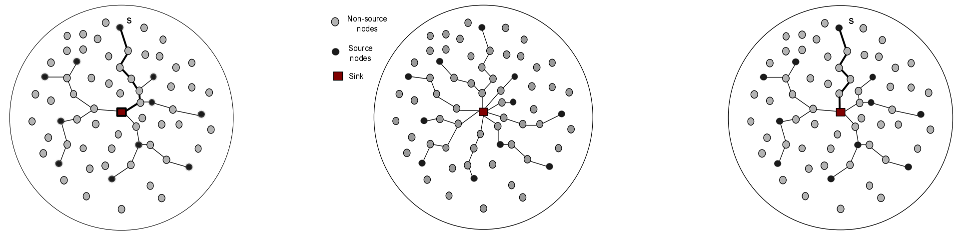
[1] A measure of the energy efficiency of a data gathering structure. Defined as the total number of transmissions required for messages from all sensor sources to reach the sink.

the information reported is correlated as it is only required to find the total number of cars and consequently determine if it is greater than K. Responses to statistical queries such as *min*, *avg*, *max*, usually fall under this category.

Characterizing the correlation existing between sensor data is a fairly complicated task, since the nature of correlation differs with the type of applications considered. Even for a simple correlation model, the mathematical representation becomes difficult when multiple distributed sources are involved. For simplicity, we adopt the same correlation model used in reference [16], where each raw data packet is assumed to bring a fixed amount of new information into the aggregated data packet. Specifically, if ρ is defined to be the correlation degree, and m to be the size of raw data packets generated by sensor nodes, then after aggregation of two data packets, the message size becomes $m + (1 - \rho)m$. Similarly, for n sources, the aggregated data packet has a size of $m + (n - 1)(1 - \rho)m$. Correlation degree $\rho = 1$ means that two messages are perfectly correlated (i.e. semantic correlation). Therefore they can be reduced to one message of the same size. Correlation degree $0 < \rho < 1$ indicates that two messages are partially correlated (i.e. spatial correlation), while $\rho = 0$ implies that two messages are independent of each other.

A data gathering structure in WSNs usually involves multiple source nodes and a sink node, hence is a tree rooted at the sink that spans those source nodes. Given correlation existing between messages collected by sensor nodes, we want to design data gathering trees that minimize energy consumption by properly aggregating sensor messages *before* they reach the sink. Since aggregation reduces the total sensor data size by eliminating the inherent redundancy, the total transmission cost can be decreased, thereby conserve energy.

The optimal data gathering tree that minimizes energy consumption of data gathering processes varies with the correlation model and degree of correlation between sensor data. For perfectly correlated sensor data where two messages can be integrated into one message of the same size, it is well-known [16] that the Steiner Minimum Tree (SMT) over all sources, sink and some of the non-source nodes gives the optimal message cost. Figure 1(a) is a Steiner Minimum Tree over a set of sources and sink. Since the message complexity of this tree is minimum, it is the optimal data aggregation tree for the given sensor nodes. However,



(a) (b) (c)

Figure 1: Optimal(a), Correlation Unaware(b), and Delay-Bounded Optimal (c) Aggregation Trees

the computation of Steiner Tree is a NP-Hard problem [49]. As the number of nodes in the network increases, the computation cost of generating SMT increases exponentially. Therefore, it is not practical to compute SMTs for large scale sensor networks. There are also some sensor applications in which the messages are only partially correlated. For such correlation models, no optimal aggregation structure has been found up until now. [16] formerly identified this problem as an original spanning tree optimization problem and proved that this problem is NP-complete too. But the authors didn't discuss how the optimal structure for perfect correlation works under non-perfect correlation model. Neither did they provide any insight about how the optimal structure should resemble under these circumstances.

Since the optimal solutions are NP-complete, many researchers have studied the problem of reducing energy consumption of data gathering process by constructing correlation aware aggregation trees that are approximations of optimal solutions. In [29], [24], and [17], simple, heuristic-based approaches are proposed to construct correlation-aware structures to achieve efficient aggregation.

While these approaches do provide energy improvement as proved, they are all centralized in nature, therefore require either complete knowledge regarding the number and locations of sources or centralized computation and explicit communications to construct and maintain the data aggregation trees. Moreover, since the optimal trees for different sets of sensor nodes are different, a dedicated tree construction process may be needed for each data gathering round with different source nodes. The overhead of those construction and maintenance processes may even offset the energy benefits of correlation aware aggregation.

On the other hand, there are also data gathering solutions that are correlation unaware (approaches that do not explicitly utilize correlation to reduce the energy consumption of data gathering process), and the most representative structure is a shortest path tree (SPT). Since the primary goal of this structure is to minimize delay, SPT does not explicitly make use of the correlation between sensor data. Several correlation unaware approaches proposed in the context of sensor networks such as directed diffusion [28] and GPSR [30] can be represented by SPTs. Figure 1(b) shows an example of a shortest path tree over the same set of sources and sink as those of figure 1(a). It is obvious from the figure that the message cost of the SPT is higher than the cost of the SMT. Even though opportunistic aggregation may possibly occur when different paths overlap with each other, SPT does not maximize the degree of aggregation possible in the network, hence structure-wise the SPT is less energy efficient.

However, a correlation unaware tree can usually be constructed without explicit coordination between the nodes and the sink. For instance, SPT trees can be established in a distributed fashion or pro-actively before the data gathering process. Furthermore, different shortest path trees for various sets of sources can be derived from the same shortest path tree over the sink and all sensor nodes by trimming branches that are entirely over non-source sensor nodes. Thus, the cost of explicit tree construction is eliminated for correlation un-aware aggregation trees.

Hence, for a data gathering tree to be energy efficient, the overhead involved in the tree construction process should also be taken into account and the energy savings of the resulting structure should be the net savings after accommodating the cost incurred in construction. Due to this reason, *correlation aware aggregation may not be the best choice if the energy improvement it enables is not significant over that of correlation unaware aggregation unless correlation aware aggregation can be achieved with low overhead.*

As a result, quantitative study of the energy gains obtained by correlation aware and unaware structures for various sensor network configurations will be desirable and insights acquired from the study can be used to guide the design of energy efficient data gathering structures in WSNs.

CHAPTER III

A QUANTITATIVE STUDY ABOUT CORRELATION AWARE AGGREGATION: THE INVESTIGATION

3.1 Research Contributions

In the first part of the research, we study the tradeoffs involved in using correlation aware structures in practical sensor applications. We specifically explore how the improvement in energy efficiency is affected by network conditions, defined by several parameters including the node density, source density, the physical distribution of sources, the correlation degree, and the delay tolerance.

Promptness of data gathering is important for many sensor applications such as target tracking and decision making. For such applications there is usually a deadline posed to each round of data gathering, after which information collected is considered useless. In this case, data aggregation structures should be carefully designed such that the real-time requirement is satisfied. At the same time, it is still desirable to reduce the cost of data aggregation as much as possible. This poses a constrained optimization problem. With delay constraints the cost of an aggregation structure may not be reduced to its fullest extent. Figure 1(c) is an illustration of this problem. Referring back to figure 1(a), if a delay bound of 5 hops is imposed on the data gathering process, the highlighted path from source s to sink is not acceptable since the delay is 6 hops. To satisfy the delay tolerance, shortest path from source s has to be used, resulting in a data gathering tree shown in 1(c), with less aggregation efficiency.

Also, the benefits of a correlation aware data gathering structure come at the expense of a construction process that typically incurs more overhead than that for a simpler structure such as the SPT, both because of the coordination required for the construction and the fact that unlike the SPT the correlation aware structure needs to change for each new set of source nodes. Hence, for a correlation aware aggregation tree to be energy efficient, the overhead involved in the tree construction should also be taken into account and the energy savings of the resulting tree should be the net savings after accommodating the cost incurred for the construction.

In this work we investigate the energy efficiency of the correlation aware aggregation process through comprehensive quantitative analysis. We specifically explore how the improvement in energy efficiency is impacted by network conditions defined by several parameters including the node density, source density, the physical distribution of sources, the correlation degree, and the delay bound. We present observations from the simulation results and draw inferences on the trade-offs involved in achieving energy efficiency.

In studying the improvements in energy efficiency with respect to specific network parameters, we also answer two fundamental questions:

1. *Is there a practical limit on the achievable improvement in energy efficiency by adopting a correlation aware aggregation structure as opposed to a correlation unaware structure?* The answer to this question will establish practical bounds on the energy efficiency improvement that can be achieved, and in turn provide a motivation or lack there-of for performing correlation aware aggregation in the first place.

2. *Is there a maximum usable delay bound that can deliver the maximum achievable energy cost improvement?* The answer to this question will establish a practical bound on how delay tolerant a WSN application needs to be in order to get the maximum energy efficiency benefit.

The contributions of this preliminary research is summarized as follows:

- We characterize through quantitative analysis how the energy improvement of a correlation aware aggregation structure is impacted by different network parameters. We show that the energy improvement tends to be bounded by a small constant under many network scenarios. Furthermore, the improvement corresponds to when the additional cost of establishing a correlation aware structure is not taken into account, in the presence of which the improvement will be further reduced.

- We characterize the energy performance of SMTs under different sensor data correlation degrees [1]. We show that SMT is a energy efficient aggregation structure only when the correlation degree is relative high. For low to medium correlation degrees, correlation structures that maintain low average aggregation delay are more energy efficient.

- We also characterize the maximum useable delay bound for achieving the maximum energy efficient structure. We show that the maximum useable delay bound is a small constant times the delay along the maximum length shortest-path in the default shortest path tree.

The rest of this section is organized as follows: In section 3.2, we describe the evaluation methodology and parameters. The optimization algorithm used for cost evaluation is also briefly introduced. In section 3.3, we present comprehensive simulation results for varying parameters as well as brief explanations of the inferences drawn from those results. In section 3.4, we highlight a major observation we made from simulation study, and derive analytical expressions to corroborate this observation. Finally we conclude this preliminary research in section 3.7.

3.2 Evaluation Model

We use a custom-built simulator written in C++ for all our simulations. The simulator takes as input the number of nodes, source density, source distribution and correlation degree, and the outputs are the respective correlation unaware aggregation trees and correlation aware aggregation trees with different delay bounds. The simulator also calculates the cost of each tree structure as another output.

[1] A measure of the degree of information redundancy between sensor messages. For two messages of the same size m, if the correlation degree is ρ, then after aggregation, the resulted combined message size is $m + (1 - \rho)m$. Hence, $\rho = 1$ means that two messages can be integrated into one message of the same size (perfectly correlated), $\rho = 0$ indicates that there is no information redundancy between the two messages, and $0 < \rho < 1$ means that the two messages are partially correlated

3.2.1 Evaluation Metrics

Most of the energy consumption in a data gathering process is due to communication. Hence, the amount of communication (number of transmissions) required is directly related to the cost of the aggregation tree. Thus, we consider the aggregation tree cost - the number of edges on a given aggregation tree - as the measure of energy efficiency of the corresponding data gathering process.

The metric we use to measure the energy efficiency improvement provided by correlation aware trees is the *cost ratio*, which is defined as the ratio of the cost of the correlation unaware tree to that of the correlation aware tree over the same set of sources and sink. The shortest path tree is constructed with the purpose of minimizing end to end delay for each source. However, when multiple paths from different sources to sink overlap at some intermediate relay nodes, opportunistic aggregation is possible. We assume such opportunistic aggregation to take place in all our evaluations. Several synchronization schemes exist to enable such opportunistic aggregation [53]. For a correlation aware tree the degree of aggregation is higher. Thus, the energy consumption of correlation aware tree tends to be lower. The cost ratio defined in the above fashion measures the relative efficiency of the aggregation aware tree to that of an aggregation unaware tree.

In most sensor applications, delay bound is typically defined to be the maximum delay instead of the average delay required to collect all sensor data. In the aggregation process, messages from sources closer to sink need to be held at some intermediate nodes until other messages from sources farther away arrive at this node in order to achieve maximum aggregation possible. For this reason, the delay incurred in the entire data gathering process is proportional to the maximum delay required to gather data from the source that is farthest from the sink.

In a data gathering process, the delay at each hop of the aggregation tree should include transmission delay, contention delay and aggregation delay. Transmission delays are typically small compared to the delay involved in aggregation and can hence be ignored. Also, for easy analysis, we assume a contention-free environment where centralized MAC layer scheduling is used to coordinate transmissions within a contention region. Therefore, the most important factor that contributes to the data gathering latency is the aggregation delay. It comprises not only of the processing time for aggregation at each node, but also the time that an aggregation node takes to wait for data from all downstream nodes in the tree to reach it. Thus, the total delay for a certain data gathering path can be assumed to be proportional to the number of hops on the path. Consequently, we specify delay constraints as the maximum allowable path length in terms of hop count.

3.2.2 Evaluation Environment and Parameters

To study the energy efficiency and tradeoffs of correlation aware aggregation trees, we consider a typical sensor network scenario where a total of n sensors are randomly distributed in a disk of radius R. All the sensors communicate using the same transmission range, which is slightly higher than that required for minimum connectivity [33]. Of the n sensors, k sensors are randomly chosen as sources to report data to the sink, which is located at the center of the disk. In this case, data aggregation trees span all sources and are rooted at the sink. This configuration is representative of many sensor network applications and results derived from it are easily extensible to other scenarios such as multiple sink applications.

The following network parameters are used for a comprehensive evaluation:

1. *Delay bound:* deadline imposed by a sensor application to one round of data gathering.
2. *Node density:* total number of nodes distributed in an unit area in the sensor network.
3. *Source density:* ratio of the number of sensors that send data packets to the sink to the total number of sensor nodes in the network.
4. *Source distribution:* geographical distribution of source nodes - uniform or non-uniform.
5. *Correlation degree:* measure of how much information two raw data packets share with each other.

3.2.3 Algorithms

We choose Steiner Minimum Tree (SMT) as the correlation aware structure, since it is the optimal aggregation structure [16] when sensor data are perfectly correlated. On the other hand, SPT is selected as the correlation unaware structure since it minimizes the delay required for data aggregation. Notice that SPT is also the most efficient aggregation tree structure when there is no correlation between sensor data. Thus, it is expected that for partially correlated sensor data gatherings, the optimal aggregation tree structure is an intermediate structure between SMTs and SPTs. Consequently, the energy improvement of SMT over SPT that we study in this paper serves as an upper bound on the energy improvement possible for all other correlation models as well.

The well-known Dijkstra's algorithm is used to compute shortest path tree in the simulations. For SMT, since it is a NP-hard problem, we resort to heuristics to generate near-optimal aggregation structures. To evaluate the impact of delay sensitivity of the application on the cost of a near-optimal tree, we need an algorithm that generates near-optimal trees for various delay constraints. Specifically, if the delay bound for a certain data gathering task is D, the delay incurred on the longest path of the near-optimal aggregation tree should be less than or equal to D. From now, *we refer to the delay-bounded near-optimal tree as DB-SMT (delay-bounded steiner minimum tree) and the near-optimal tree without delay bound as simply SMT.*

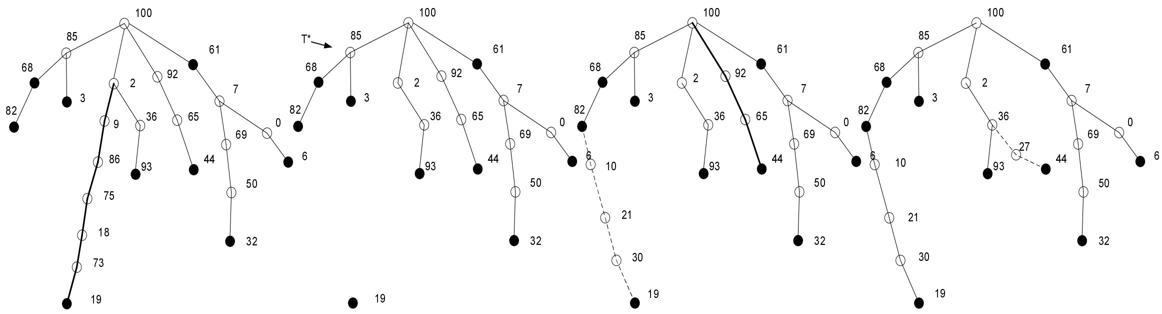
A set of algorithms developed in the context of multicast applications can be used for this purpose. Most multicast routing algorithms are designed to support large number of simultaneous multicast sessions efficiently. A multicast tree that minimizes the total bandwidth utilization of the network links is established from the source to destinations in these algorithms. Hence, these algorithms can be used for sensor network aggregation, with the only difference being that the data flows in sensor networks are in the reverse direction. Some of these algorithms are specifically tailored to multicast applications that are delay sensitive, such as multimedia streaming. Such algorithms, called Constrained Steiner Tree heuristics (CST), generate minimum cost multicast trees within certain delay constraints and can hence be used exactly for our purpose.

We choose an approximation algorithm called BSMA (bounded shortest multicast algorithm) to generate the DB-SMTs. This algorithm has been proven to be able to construct DB-SMTs with additional costs less than 7% than that of the optimal Steiner Minimum

Tree, and has been shown to achieve lower costs than other related strategies [45].

In the following paragraphs we briefly introduce the BSMA algorithm, as well as how we use it in the WSN context. The BSMA algorithm starts with a minimum delay tree (using any shortest-path algorithm) and iteratively improves the cost of the delay constrained tree employing a multi-pass approach. For each step, the delay bound is always satisfied. During the optimization process, tree costs are decreased monotonically, and the iterative process terminates when no more cost reduction is possible. Let T_0 denote the initial shortest path tree. At iteration step j, the tree T_j is transformed to tree T_{j+1} with the following procedure:

- choosing the path to be taken out of T_j and obtaining two disjoint subtrees T_j^1 and T_j^2;
- finding a new path to connect T_j^1 and T_j^2, resulting in the new tree T_{j+1} with smaller cost, while the delay bound is satisfied



(a) *Step*1 (b) *Step*2 (c) *Step*3 (d) *Step*4

Figure 2: BSMA Optimization Process

A candidate path in tree T_j for cost improvement is called a superedge. It is a simple path such that all internal nodes on the path are non-source nodes. For each iteration step, the longest superedge on the tree is deleted and replaced with an alternative path with lowest cost on the condition that after the replacement the delay constraint is not violated. To illustrate this procedure, we take a small shortest path tree as shown in figure 2(a). In this figure, black nodes on the tree T_j are source nodes and white nodes on the tree are non-source relay nodes. Sink is the node with id 100. For the first step, the superedge

"$2 \rightarrow 9 \rightarrow 86 \rightarrow 75 \rightarrow 18 \rightarrow 73 \rightarrow 19$" is identified as the longest superedge and deleted, resulting in the subtree T_j^* and node 19. To find the best replacement, every path between source node 19 and all nodes on tree T_j^* is sorted out and checked for delay violation. Among those paths that satisfy the delay bound, the one with lowest cost "$19 \rightarrow 30 \rightarrow 21 \rightarrow 10 \rightarrow 82$" is selected to generate an updated tree T_{j+1}. This new superedge doesn't have to be tested again since it is already optimized. Among the remaining superedges, "$100 \rightarrow 92 \rightarrow 65 \rightarrow 44$" is the longest one, and after similar process, "$36 \rightarrow 27 \rightarrow 44$" replaces this superedge. The cost of tree T_{j+1} is 1 hop less than the cost of tree T_{j+2}, but the delay from node 44 increases from 3 to 4 hops. This optimization procedure continues until no more superedges can be further replaced.

3.2.4 Methodology

To study the energy efficiency and tradeoffs in WSN data aggregation process, we start from a shortest path tree spanning the sink and all the source nodes in the network, and apply the BSMA algorithm to reduce the cost of the tree. Using this algorithm, different tree structures can be obtained for different delay constraints. The maximum hop count in the initial shortest path tree is taken as the lowest possible delay bound and the DB-SMT tree is initially generated with this delay bound. The delay bound is then relaxed to obtain DB-SMT trees with further reduced costs.

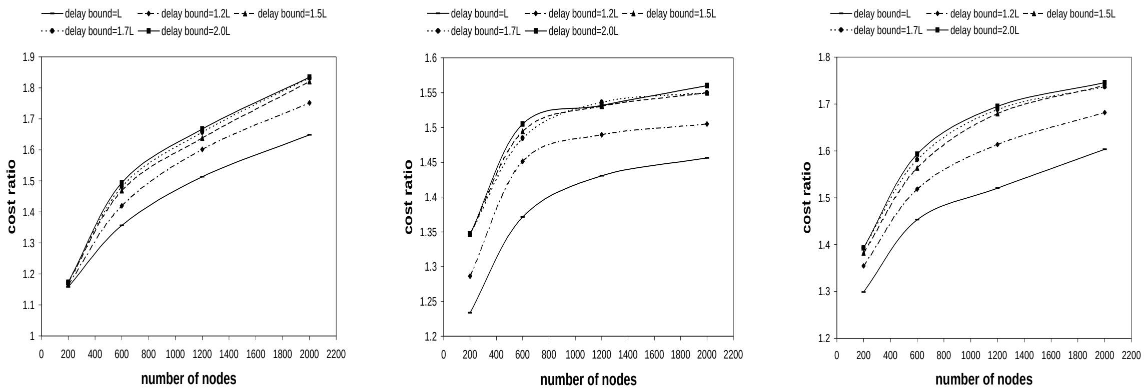
For each network configuration, we vary the network parameters specified in the previous subsection, generate SPT and DB-SMT trees for each configuration, and take the cost ratio between SPTs and DB-SMTs. Each graph shown below contains several curves displaying the relationship between cost ratio and one of the varying network parameters, with each curve corresponding to a specific delay constraint. The simulation results and inferences are identified and presented in the following section. However, detailed analysis and explanations are not included in this proposal due to space limitation. Please refer to [57] for further details.

3.3 Performance Results

In this section we present simulation results to show the energy-delay efficiency of aggregation trees under various network conditions.

3.3.1 Varying Node Density

Figure 3 shows the cost ratio of SPT vs. DB-SMT when the source densities are 1/20, 1/10 and 1/5, and the number of sensor nodes distributed in the field (n) is increased from 200 to 2000.



(a) *source density* $= 1/20$ (b) *source density* $= 1/10$ (c) *source density* $= 1/5$

Figure 3: Performance Improvement over SPT for Different Node Densities

It can be observed from the results that the cost ratio between SPT and DB-SMT increases with node density. This implies that correlation aware data gathering is more efficient when the density of sensor nodes is large. This can be intuitively explained as follows: with high node density, the probability of shortest paths over-lapping with each other is low; hence, SPT has very low aggregation efficiency and there is greater potential for energy improvement using a DB-SMT. Consequently, the cost ratio improves as node density increases. To further illustrate this observation, we plot the structure of SPT and DB-SMT trees constructed when n is 400 and 1200 in Figure 8. In both configurations, the number of source nodes is 1/10 that of total sensor nodes. For the case of $n = 1200$, it can be seen that many parallel shortest paths exist in the SPT structure. However, after optimization for aggregation, most of these separated paths are combined, thereby enabling great cost savings in DB-SMT. However, for the $n = 400$ case, SPT is already an efficient structure in terms of path sharing. Thus, the improvement after optimization is not significant.

We also observe that for different source densities, the increasing trend of cost ratio remains to be the same. However, the absolute value of cost ratio reduces as source density increases. Further, for the same source density, when the delay constraint is increased, the cost ratio between SPT and DB-SMT increases, but the increase in ratio tends to saturate when the delay bound is more than 1.5 times the longest shortest path length. The specific reasons for these observations will be presented when the impact of source density and delay constraints are investigated subsequently.

Although the cost ratio increases with node density, we can observe that it tends to saturate when node density is high (corresponding to $n = 1200$ in our simulation settings). This implies that in contrast to common belief, energy improvement of DB-SMT over SPT does not scale significantly with node density. Theoretical analysis that provides a tight bound on the rate at which the cost ratio improves with node density will be presented in Section 3.4.

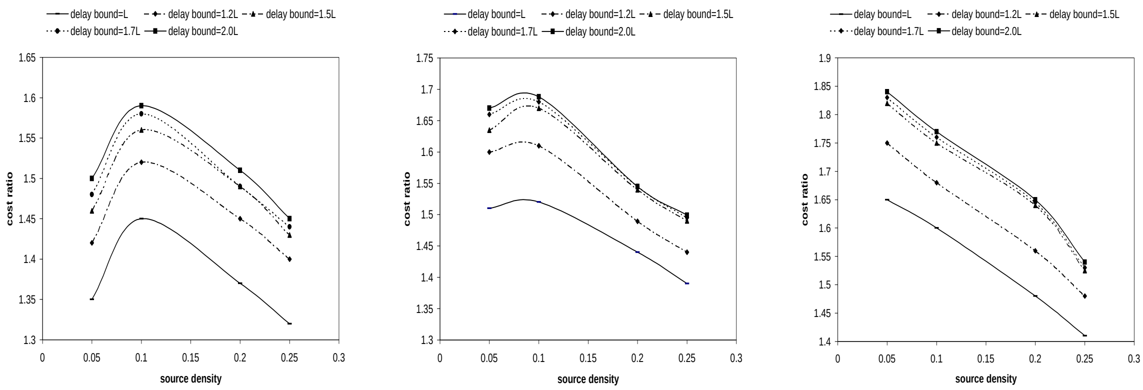
The results also indicate that at low node density, correlation aware data gathering does not bring significant cost improvement. If we require a correlation aware aggregation tree to provide a factor of at least 1.5 times in cost improvement (50% improvement), then from the results it is clear that this is possible only when the node density is sufficiently high with $n > 600$. This in turn implies that correlation aware aggregation does not provide desired energy efficiency for low node densities.

Thus, from the study of cost ratio variation with node density, we have the following insights:

Cost ratio of SPT over DB-SMT increases with node density in sensor networks, but tends to saturate with increasing node density. For correlation aware aggregation trees to achieve a desirable energy improvement, the node density of the sensor network should be relatively high.

3.3.2 Varying Source Density

To investigate how the density of source sensor nodes affects the efficiency of aggregation tree, we compare the cost of SPT and DB-SMT across a range of source densities and node densities. Figure 4 shows how cost ratio of SPT and DB-SMT varies with source densities when $n = 600$, $n = 1200$ and $n = 2000$. Each curve consists of four data points with respect to source densities of 1/20, 1/10, 1/5 and 1/4.



(a) $n = 600$ (b) $n = 1200$ (c) $n = 2000$

Figure 4: Performance Improvement over SPT for Different Source Densities

It can be observed that when node density is low, the cost ratio increases with source density, reaches a maximum, and then starts to decrease again. However, for high node density, the cost ratio decreases monotonically with source density. When there are fewer sensor nodes in the network, due to the relatively small SPT cost, the possible cost reduction achievable from optimization in DB-SMT is limited. On the other hand, when source density is higher than 1/5, a considerable fraction of nodes on SPT are sources, implying that SPT is already an efficient structure. Consequently, the possible cost reduction from optimization in DB-SMT is once again less. An important factor that determines the degree of cost improvement is the inefficiency of the SPT structure. Thus, at very low and very high source densities, the higher efficiency of SPT structures reduces the cost ratio improvement, resulting in a peak value at an intermediate value of source density.

The inference with respect to node density is the same as before. When the node density is high, the path diversity in SPT also increases. Thus, the shortest paths in SPT diverge from each other even at low source densities, leaving considerable margins for cost improvement in DB-SMT. This results in the monotonically reducing cost ratio at $n = 2000$.

Simulations with source densities larger than 1/4 were also conducted, and cost ratios were observed to be less than 1.2. This can also be extrapolated from the trend of the curves in Figure 4. From the study of cost ratio variation with source density, we obtain the following insights:

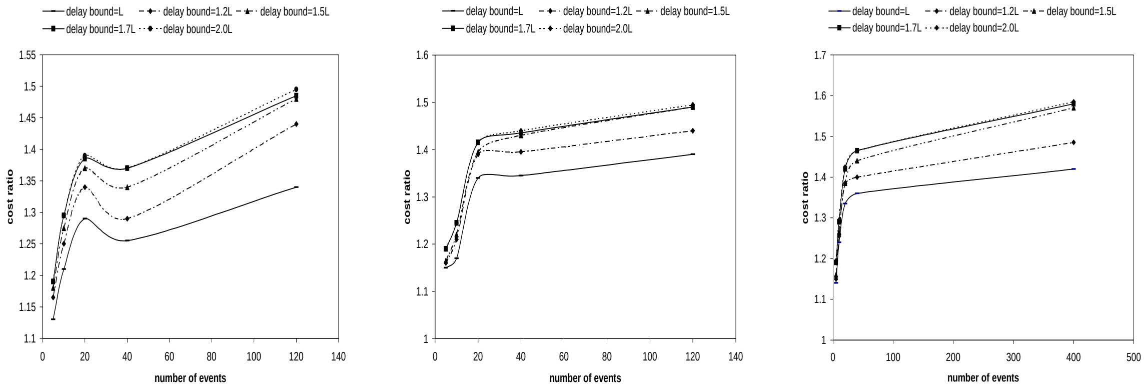
Cost ratio of SPT over DB-SMT decreases with increasing source density when node density is high. However, with low node density, medium source density ensures the best possible cost improvement.

3.3.3 Varying Source Distribution

In the previous discussions, we had assumed that sources are uniformly distributed in the network. However, this may not always be the case in sensor network applications. There are situations where only certain specific locations (where scattered events occur) in the network need to be monitored, in which case the sink gathers data from sensor nodes around these events. Under these circumstances, source nodes can no longer be considered to be uniformly distributed. In this subsection, we study how the distribution of sources affects the effectiveness of correlation aware aggregation.

In this set of simulations, n increases from 200 to 2000, and each configuration has a total of $s = n/5$ sources distributed in the network. The number of events e (locations) in the network for each scenario increases from 5, 10, 20, 40 to s. Sources are equally distributed in the different event locations. When the number of events is 5 and 10, the sources are highly cluttered around the event locations, and as event number increases, the source distribution becomes closer to uniform distribution. This model is similar to the event-radius model used in [13].

From the results presented in Figure 5, it can be seen that the cost ratio increases with the number of events. The overall trend of the cost ratio improvement can be explained as follows. The sources tend to be densely distributed around event locations when there are few events in the network. The shortest paths from the same event location to the sink



(a) $n = 600$ (b) $n = 1200$ (b) $n = 2000$

Figure 5: Performance Improvement over SPT for Different Source Distributions

can combine with each other at an early stage, thereby making SPTs inherently efficient in terms of path sharing. This can be observed from Figure 5, where the cost ratio is less than 1.3 when the number of events is 5 and 10. However, the path diversity of SPTs tends to increase as the number of event locations increases with the source distribution tending towards uniform distribution. Consequently, the cost reduction for correlation aware data aggregation becomes greater.

From the study of cost ratio variation with source distribution, we gain the following insight:

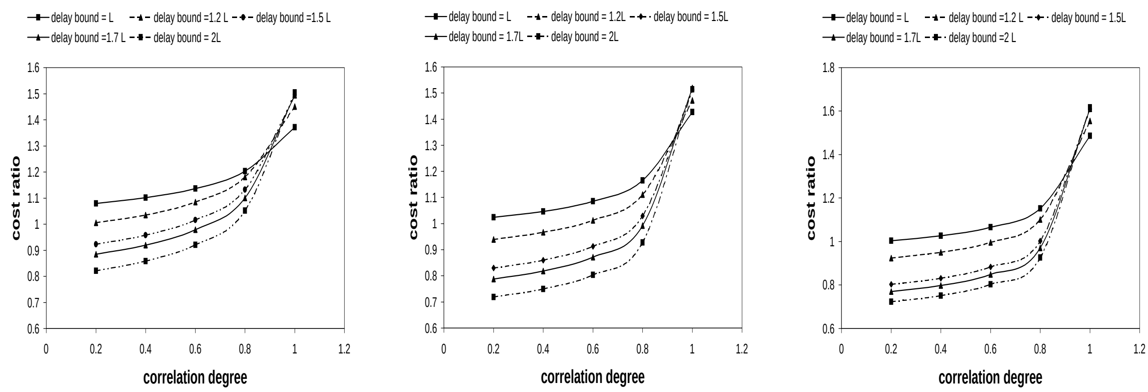
Cost ratio of SPT over DB-SMT increases as the distribution of source nodes tends towards uniform distribution

3.3.4 Varying Correlation Degree

It is possible that the data gathered in certain sensor network applications are not perfectly correlated, in which case the correlation degree ρ will be less than one. Several works [16,38] have studied this problem before. However, none of them have identified the effectiveness of SPT versus SMT with respect to varying correlation degree.

Characterizing the correlation existing between data collected in sensor networks is a fairly complicated task, since the nature of correlation differs with the type applications considered. Even for a simple correlation model, the mathematical representation becomes difficult when multiple distributed sources are involved. [38] presents a correlation model that uses joint entropy to define correlation between two sources. A constructive technique is also proposed to characterize correlation when multiple sources are involved, but the calculation becomes intractable when there are large number of sources, uniformly distributed in a 2-dimensional field. For simplicity, we adopt the same correlation model used in [16], where each data packet is assumed to bring a fixed amount of new information into the aggregated data packet. Specifically, if ρ is defined to be the correlation degree, and the sizes of raw data packets generated by sensor nodes is defined to be m, then after aggregation of two data packets, the message size becomes $m + (1 - \rho)m$. Similarly, for n sources, the aggregated data packet has a size of $m + (n - 1)(1 - \rho)m$.

In this set of simulations we choose the number of nodes to be 600, 1200 and 2000; source density to be 1/5; and delay constraints of L, $1.2L$, $1.5L$, $1.7L$ and $2.0L$. Figure 6 illustrates how cost ratio varies with the correlation degree and delay constraints.



(a) $n = 800$ (b) $n = 1200$ (b) $n = 2000$

Figure 6: Performance Improvement over SPT for Different Correlation Degrees

It can be seen that the cost ratio increases with correlation degree. This trend remains the same for all node densities. However, for higher node densities, the cost improvement of DB-SMT over SPT is lower. We explain the overall increasing trend of cost ratio with increasing correlation degree as follows: when $\rho \to 0$ (raw data packets are un-correlated with each other), SPT is the optimal structure since aggregation does not help reducing the transmission and hence the energy cost. Thus, the best approach is to deliver each message along the shortest possible route to the sink. On the other hand, when $\rho \to 1$, SMT is the optimal structure with respect to energy efficiency, as established earlier. Therefore, we expect the optimal structure to be close to SPT for small correlation degrees, wherein progress towards the sink is more important than en-route aggregation. Due to this reason, the cost ratio of SPT over DB-SMT increases with increasing correlation degree.

In each of the results, it can be seen that for most of the correlation degrees, DB-SMT with a lower delay bound results in a higher cost ratio than DB-SMT with a higher delay bound. However, this trend is completely reversed when $\rho = 1$. Also notice that when delay bound is higher than $1.5L$, the cost ratio between SPT and SMT is less than one for some of the lower correlation degrees. This in turn implies that SPT is a more efficient structure for aggregation than DB-SMT under those circumstances. These trends are counter-intuitive, because it is expected that higher delay bounds assist in path sharing and thereby energy cost reduction in DB-SMT by traversing as many nodes as possible at an early stage of the aggregation path. However, results indicate that increasing the delay constraint and thereby extending the path for more aggregation does not improve energy efficiency for most of the partially-correlated ($\rho < 1$) cases. The reasoning for this observation is as follows.

When the simulation results were further analyzed, it turned out that as the maximum path length (delay constraint) increases, the average path length for a DB-SMT also increases. For example, when delay bound was 10 hops, the corresponding average path length was 7.8 hops. However, when the delay bound was 20 hops, the average path length increased to 10.2 hops. The average path length of an aggregation tree has two conflicting impacts on its energy efficiency. On one hand, the smaller the average path length, the lesser the number of hops (transmissions) towards the sink and hence lower energy cost. On the other hand, a shorter average path length also implies lesser room for aggregation, leading to a lower energy efficiency. The relative impact of the two components and the net resulting impact on energy efficiency is in turn dependent on the correlation degree of the sensor data. For lower correlation degrees, the room for aggregation is inherently low. Hence, a shorter average hop length would help reduce the energy cost. Due to this reason, DB-SMTs with lower delay constraints that facilitate explicit aggregation while at the same time maintaining smaller average path length perform the best. However, at high correlation degrees, the larger room for aggregation and hence cost reduction overcomes the additional cost due to increased average hop length, resulting in DB-SMTs with larger delay constraints performing the best. These observations and results clearly indicate that SMT serves to be the optimal aggregation structure only when data from different sources are highly correlated. For scenarios where correlation between the sensor data is low, SPT or DB-SMT with lower delay bound is a better structure for energy efficiency.

These observations and results clearly indicate that SMT serves to be the optimal aggregation structure only when data from different sources are highly correlated. For scenarios where correlation between the sensor data is low, SPT or DB-SMT with lower delay bound is a better structure for energy efficiency.

From the study of cost ratio variation with correlation degree, we obtain the following insights:

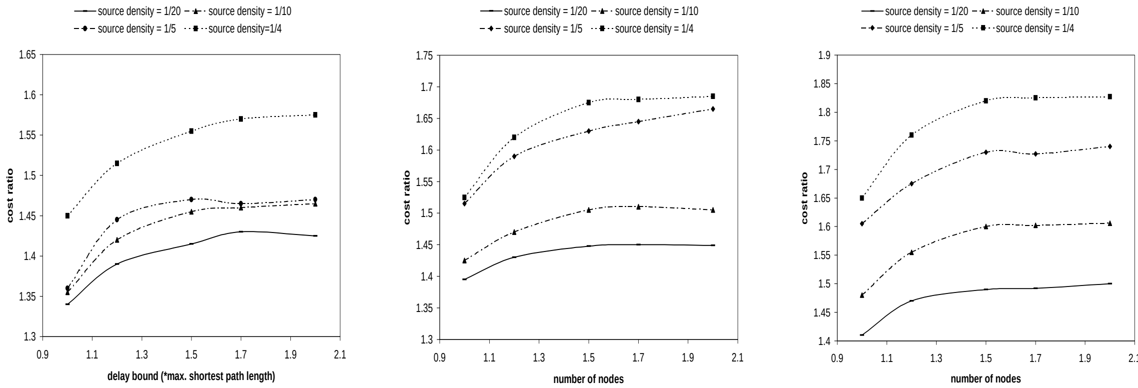
Energy efficiency of DB-SMT increases with correlation degree. DB-SMT with the lowest delay bound proves to be the most energy efficient for low to moderate correlation degrees. Higher delay bounds help improve aggregation efficiency only when the correlation degree is relatively high. The high correlation degrees also ensure the optimality of the SMT structure.

3.3.5 Varying Delay Bounds

One of the objectives of this work is also to understand the limit of data gathering delay bounds on the energy efficiency of correlation aware aggregation trees. In all the results discussed thus far, we present curves corresponding to delay bounds (D) L, $1.2L$, $1.5L$, $1.7L$, and $2.0L$. To study the variation of cost ratio with respect to delay bounds in depth, results from some of the simulations ($\rho = 1$) are re-plotted in Figure 7.

It is clear from figure 7 that the cost ratio increases with increasing delay bounds, which indicates that less restrictive delay tolerance helps maximize en-rounte aggregation and hence the cost efficiency.

Higher delay bounds imply that the aggregation path can be longer in order to maximize en-route aggregation. Both $D = 1.2L$ and $D = 1.5L$ result in significant cost improvement over $D = L$ scenario. However, the growth of cost ratio slows down and tends to saturate

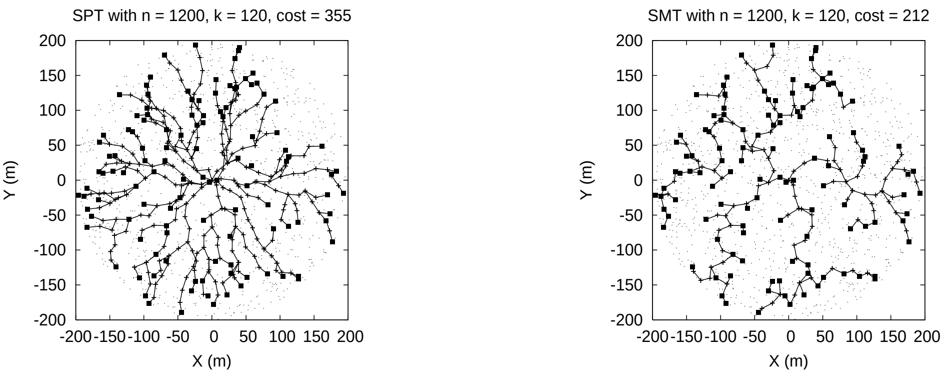


(a) $n = 600$ (b) $n = 1200$ (c) $n = 2000$

Figure 7: Performance Improvement over SPT for Different Delay Bounds

after $D = 1.7L$. This is a very interesting observation. Generally speaking, the intuition is that the longer a path is, the more data packets can be aggregated en-route. Thus, higher delay bounds allow the creation of aggregation trees with lower cost. But simulation results show otherwise: aggregation path longer than twice the longest shortest path do not help significantly in reducing the cost.

However, the growth of cost ratio slows down and tends to saturate after $D = 1.7L$. This is a very interesting observation.



(a) $SPT, n = 1200$ (b) $SMT, n = 1200$

Figure 8: Topology of SPTs and SMTs

To understand this phenomenon better, we plot the structures of SMT and SPT for $n = 1200$ in figure 8 (a) and (b) respectively. It can be seen from the two structures that the "backbone" structure of SMT is similar to that of SPT, where several shortest paths tend to divide the network graph uniformly. The difference is that there are lesser number of shortest paths in the backbone of SMT. Sources not on the "backbone" are connected by "branches" to the backbone structure. Thus, while this structure is more efficient than SPT in terms of cost, with the paths being combined as much as possible, the longest path length is not significantly higher than that in the SPT structure. Further, the maximum delay tolerance that is helpful in reducing the aggregation tree cost is given by the length of the longest path on the SMT structure. Hence, given the practical structural characteristics of SMT, the longest path in SMT tends to be only a small constant order when compared to that of the longest path in the SPT structure.

However, notice that all the above discussions are pertaining to $\rho = 1$ correlation model. If the correlation degree of sensor data is low, then a lower delay bound would yield a better performance. Hence, the cost ratio trend would reverse in that case for low correlation degrees.

From the study of cost ratio variation with respect to delay bounds, we have the following insight:

The cost ratio of SPT over DB-SMT increases as delay bound increases for high correlation degrees and tends to saturate. Delay bounds beyond twice the maximum shortest path length do not help reduce the DB-SMT cost further.

3.3.6 Summary

In this subsection, we summarize all the observations and insights derived from simulation studies.

- We have shown that the cost ratio of SPT over DB-SMT increases with node density in the sensor network, but tends to saturate as the node density increases further.

- Further, when node density is high, the cost ratio of SPT over DB-SMT decreases with increasing source density. However, at low node density, a moderate source density delivers the best cost improvement.

- With respect to the impact of source distribution on aggregation efficiency, we observe that cost ratio of SPT over DB-SMT increases as the distribution of source nodes tends to uniform distribution.

- The energy delay tradeoff of correlation aware and unaware tree can be summarized as follows: The cost ratio of SPT over DB-SMT increases as delay bound increases for high correlation degrees. Delay bounds beyond twice the maximum shortest path length do not help reduce DB-SMT cost further. Furthermore, the cost ratio tends to decrease as delay bound increases for low correlation degrees.

- For different correlation models, we find that the energy efficiency of DB-SMT increases with correlation degree, and DB-SMT with the lowest delay bound is the most energy efficient for low to moderate correlation degrees. Higher delay bounds help improve aggregation efficiency only when correlation degree ρ is sufficiently high. The high correlation degree also ensures the optimality of SMT.

Finally, we highlight three major observations we inferred from simulation study:

1. The cost ratio of SPT over DB-SMT scales very slowly (tends to saturate) with respect to node density.
2. Increasing delay bound beyond a (small) constant order of the longest shortest path length does not help reduce aggregation tree cost further.
3. SMT is a cost efficient aggregation structure only when the correlation degree is relatively high

3.4 Analytical Reasoning

In this section, we theoretically substantiate the slow rate of growth of the cost ratio of SPT over SMT with respect to node density. Specifically, we show that the expected (energy) cost improvement obtained by a SMT over SPT scales very slowly (as $\sqrt{\log n}$) with node density.

Before going through the details of the proofs, we present the details of the SPT and SMT structures considered in estimating the costs.

3.4.1 Expected SPT Cost

We consider a network graph where nodes are uniformly distributed in a unit area disk and the root of the SPT tree is at the center of the disk. For the convenience of analysis, we divide the network into layers of concentric rings, each ring consisting of all the nodes that are at the same distance (in terms of hops) away from the sink, i.e. nodes in between the i^{th} and $(i-1)^{th}$ rings are assumed to be i hops away from the sink. The distribution of nodes and sources are assumed to be uniform in the network. The uniform distribution of nodes assumed in the network corresponds to a poisson point process with a certain rate λ. A property of such a poisson point process is that the expected number of nodes in a certain subregion with area A is equal to $A * \lambda$. Hence, the expected number of nodes that are i hops away from sink increases with i^2.

In a SPT structure, each source is connected to the sink which is located at the center of the unit disk. For sources which are further away from the sink, the shortest paths can be considered to be independent of each other with a high probability. However, at a certain distance away from the sink, all shortest paths tend to converge, and the nodes within this distance belong to at least one of the shortest paths with a high probability. Hence, we assume that there exists a threshold distance and hence ring i^*, such that for all $i \leq i^*$, all nodes on ith ring are part of the SPT structure. However, for rings beyond ring i^*, only some of the nodes on each ring will be part of the SPT structure.

Based on the SPT structure defined above, we define a relaxed SPT structure SPT_r, such that the cost of SPT_r is higher than that of SPT. This relaxed SPT structure also consists of two components. The first component (SPT_0) is a SPT spanning all the nodes within i^* hops from the sink. And the second component (SPT_1) is a set of independent shortest paths such that each path connects exactly one source to a leaf on SPT_0. In other words, paths on SPT_1 never overlap with each other. Thus, the cost of SPT_r is always higher than the cost of SPT.

Now, let m be the hop number of the longest shortest path in the network, n and s be the total number of nodes and sources in the network respectively.

The expected cost of SPT_r (C_{spt}) is given by

$$E[C_{sptr}] = E[C_1] + E[C_2] \quad (1)$$

Since all nodes in SPT_0 component are part of the SPT structure, the net cost of C_1 is contributed by the number of nodes in the SPT_0 component, which in turn is given by the number of nodes on the i^*th ring. According to the property of poisson point process for uniform node distribution, we have,

$$E[C_1] = \sum_{1}^{i^*} E[n_j] = \frac{i^{*2}}{m^2} n \quad (2)$$

To obtain $E[C_2]$, we condition the product of the number of sources present in a ring j ($i^* < j \leq m$) along with their shortest distance to a leaf in SPT_0. This results in,

$$E[C_1] = \sum_{j=i^*+1}^{m} s_j d_j = \frac{m^2 - i^{*2}}{m^2} s \times \frac{2}{3}(m - i^*) \quad (3)$$

$$= \frac{2}{3}\frac{s}{m^2}(m - i^*)(m^2 - i^{*2}) \quad (4)$$

where, $\frac{m^2 - i^{*2}}{m^2} s$ is the total number of sources on SPT_1, and $\frac{2}{3} * (m - i^*)$ is the expected length of shortest paths on SPT_1.

Thus, the expected cost of the relaxed SPT structure from the costs of the two components (C_1 and C_2) is now given by,

$$E[C_{sptr}] = \frac{i^{*2}}{m^2} n + \frac{2}{3}\frac{s}{m^2}(m - i^*)(m^2 - i^{*2}) \quad (5)$$

The radius of transmission and hence the hop length is the minimum connectivity range defined in [33],

$$r = R\sqrt{\frac{\log 10n}{n}} \quad (6)$$

where R is the radius of the entire network.

Let n_{i^*} represent the total number of nodes on i^{*th} ring (with i^* hops), and s_j denote the number of sources on j^{th} ring. Since the shortest paths on SPT_1 are independent, each node on i^{*th} ring is connected to at least one shortest path on SPT_1. Therefore, we have:

$$
\begin{aligned}
n_{i^*} &< \sum_{j=i+1}^{m} s_j && (7)\\
\Rightarrow (2i^*-1)\frac{n}{m^2} &< \frac{m^2-i^{*2}}{m^2}s && (8)\\
\Rightarrow si^{*2}+2n_{i^*}-n-m^2s &< 0 && (9)
\end{aligned}
$$

Solving the above inequality, we get i^* which is given by

$$
\begin{aligned}
i^* &= \frac{-2n \pm \sqrt{4n^2+4s(n+m^2s)}}{2s} && (10)\\
&\simeq \frac{-2n \pm 2ms}{2s} && (11)\\
&\simeq m-\frac{n}{s} && (12)
\end{aligned}
$$

When the fraction of sources is large such that, $\frac{s}{n} > \frac{1}{m}$.

m can be approximated as

$$
\begin{aligned}
m &= \frac{R}{r}\beta && (13)\\
&= 1.32\sqrt{\frac{n}{\log 10n}} && (14)
\end{aligned}
$$

where R is the radius of the network, r is the minimum transmission range for connectivity defined in [33] and β is a constant. This constant is introduced to account for a path, connecting a furthermost node to sink, not being a straight line. Plugging in the transmission range defined with (6), we get,

$$
m = 1.32\sqrt{\frac{n}{\log 10n}} \qquad (15)
$$

3.4.2 Expected SMT Cost

Determining the cost of SMT in a network graph directly is rather difficult. However, for the sensor network environment considered, we can translate the cost of SMT in Euclidean space (ESMT) (whose cost is known directly) into the cost of SMT in network graphs (NSMT).

Lemma 1: The expected cost ESMT is $\Theta(R\sqrt{s})$.

Proof: From [54], the cost of a minimum spanning tree in Euclidean space (EMST) has the following upper bound:

$$E[C_{EMST}] \leq 0.707\sqrt{s}R + o(\sqrt{s}) \tag{16}$$

and has the following lower bound:

$$E[C_{EMST}] \geq \frac{1}{2}R\frac{s-1}{\sqrt{s}} \tag{17}$$

Combining the two bounds, we have:

$$E[C_{EMST}] = \Theta(R\sqrt{s}) \tag{18}$$

On the other hand, it is shown in [18] that the cost ratio of EMST over ESMT for the same set of source nodes is bounded by a small constant:

$$\frac{E[C_{EMST}]}{E[C_{ESMT}]} < \frac{2}{\sqrt{3}} \tag{19}$$

Therefore, we have

$$E[C_{ESMT}] = \Theta(R\sqrt{s}) \tag{20}$$

Lemma 2: The expected cost of NSMT is $\Theta(m\sqrt{s})$ *for the sensor network considered in network graphs.*

Proof:

Note that, the distance between two sources on ESMT can be translated into hop count directly via the following relationship:

$$H = \lceil \frac{L}{r} \rceil \tag{21}$$

where H is the hop count of path between two nodes, and L is the Euclidean distance. Since such a translation maintains the order of the cost, and m is the equivalent of R in network space, we have

$$E[C_{NSMT}] = \Theta(m\sqrt{s}) \tag{22}$$

Accordingly the expected cost of a SMT in network graph can be approximated as:

$$E[C_{smt}] = c\sqrt{s}m, \tag{23}$$

where c is a constant.

3.4.3 Cost Ratio

Proposition 1: The expected cost improvement of SMT over SPT in sensor network graph increases as $\Theta(\sqrt{\log n})$, where n is the total number of nodes in the sensor network, and s is $\Theta(n)$.

Proof: Combining equations 5 and 23, and observing the fact that the SPT_r structure considered for the analysis is a relaxed variant of the actual SPT structure, we obtain the ratio of the expected costs of the SPT and SMT structures as,

$$\begin{aligned} Cost\ Ratio &\leq \frac{E[C_{sptr}]}{E[C_{smt}]} && (24) \\ &= \frac{\frac{i^{*2}}{m^2}n + \frac{2s}{3m^2}(m - i^*)(m^2 - i^{*2})}{cm\sqrt{s}} && (25) \end{aligned}$$

where m and i^* are computed using equation (15) and (10). Plugging in m and i^*, we get:

$$\begin{aligned} E[C_{sptr}] &= \Theta(\sqrt{n \log n}) + \Theta(n) && (26) \\ &= \Theta(n) && (27) \end{aligned}$$

and

$$E[C_{smt}] = \Theta(\frac{n}{\sqrt{\log n}}) \tag{28}$$

Combining the above two equations, we get:

$$Cost\ Ratio = \Theta(\sqrt{\log n}) \tag{29}$$

From the above analysis, we arrive at the conclusion that the cost ratio of SPT over SMT increases only with $\Theta(\sqrt{\log n})$ for large n. This increase rate is responsible for making the cost ratio improvement saturate at high node densities in the simulations. Hence, we can expect that when n is sufficiently large, the energy improvement of SPT over SMT tends to saturate. Although theoretically speaking, the cost ratio still increases as a function of n, practically the improvement in energy efficiency provided by such a slow increasing rate is negligible beyond a certain node density. Consequently, for large scale sensor networks, the energy improvement of correlation aware aggregation trees is not as significant as normally expected. We discuss the practical implications of this observation in the next section.

3.5 Practical Implications

3.5.1 Practical Implications of Limited Energy Improvement

As inferred in Section 3.3, the energy improvement of correlation aware tree structures over correlation unaware tree structures is bounded by a small factor of two for all the scenarios we simulated. We have also shown through analysis that cost ratio increases as $\Theta(\sqrt{\log n})$. This indicates that even for a large scale sensor network with node densities greater than those simulated in this work, the perceivable energy improvement will still be limited due to the slow growth rate of energy improvement.

This observation implies that correlation aware aggregation may not always be a good choice for sensor data gathering. As discussed in section 1, explicit communication is required for setting up correlation aware aggregation trees. Even the approximation algorithms for constructing a Steiner tree impose some requirements such as the information regarding the number of sources and their locations to be available at the sink *a priori*. This information could potentially be obtained if the sink adopts a two-phase querying procedure, where the query is sent in the first phase and the responses collected reveal the location of the sources interested in responding to that query. In the second phase, the sink initiates the construction of the approximation of a Steiner tree to optimize the message complexity of the data sent. However, the overhead involved in determining the number and location of sources and the construction of the aggregation tree is $O(k)$, where k is the number of sources which might be comparable or even exceed the message complexity of the aggregated messages.

Furthermore, for highly dynamic sensor applications where sources change rapidly with time, the overhead of tree construction may offset the energy benefits resulting from correlation aware aggregation. It is also possible that the source nodes that are going to report data packets to sink are not known a priori, in which case correlation aware aggregation trees cannot be computed before the data gathering process.

On the contrary, correlation unaware trees such as shortest path trees can usually be established in a distributed fashion or pro-actively before the data gathering process. Furthermore, different shortest path trees for various sets of sources can be derived from the same shortest path tree over the sink and all sensor nodes by trimming branches that are entirely over non-source sensor nodes. The cost of explicit tree construction is eliminated for correlation unaware aggregation trees. Thus, given the cost and feasibility issues involved in the construction of correlation aware trees and the moderate energy improvement possible, correlation unaware approach may be a more desirable choice under many circumstances.

3.5.2 Practical Implications of the Impact of Correlation Degree

Simulations and analysis also indicate that SMT is the optimal aggregation structure only when the degree of correlation among sensor messages are relatively high. For medium to low correlation degree, DB-SMT with lowest delay bound delivers better performance. This indicates that structure-wise, long aggregation paths are not favorable choices for medium to low correlation degrees since the energy savings achieved by aggregation are limited in these cases. Hence under such circumstances, the primary concern is to deliver each message via the shortest route, and at the same time facilitate en-route aggregation as long as the resulted path lengths do not increase too much. Therefore, for sensor applications with low to moderate correlation degrees, the ideal aggregation structures should be similar to SPTs with less path diversity.

3.5.3 Practical Implications of Limited Delay Tolerance

We also observed that increasing delay tolerance does not always help reduce the aggregation tree cost. Beyond a certain threshold which is comparable to the longest shortest path length, the cost ratio improvement tends to saturate. This is because increasing aggregation path length beyond this point does not increase the aggregation efficiency much. Practically, this implies that an application does not have to be designed with large delay tolerances to ensure close to maximum energy efficiency.

3.6 Related Work

In this section, we discuss related works that have done similar studies as that presented in this thesis. For each related work, we explain its scope of study and introduce observations and results made by the authors. The similarities and differences between their results and ours are compared, and reasons for those differences are identified.

3.6.1 Related Works on Correlation Aware Aggregation Trees

[16] provided in-depth discussions related to efficient data gathering structure, and proved that the generation of an optimal aggregation structure is a NP-complete problem. Two heuristics: leaves deletion algorithm and SPT/TSP balanced tree algorithm are proposed to approximate the optimal tree and the cost ratios of approximation trees over SPTs are presented in this paper. The results presented in this paper are similar to ours in that the cost ratio of optimized tree over SPT is bounded by 0.5. This paper made a similar observation to ours that SMT resembles a combination of a SPT core and TSP paths in the outskirts. A SPT/TSP balanced tree is proposed as an approximation of optimal tree according to this observation. For an aggregation tree, SPT is built for nodes within a radius $q(\rho)$ from the root, and for the rest of the nodes, TSP paths are used to connect sources in a certain subregion to the existing shortest path tree. The advantage of this algorithm is that correlation degree is taken into account during tree construction process. But this structure is not adaptive to number of nodes in the network. As illustrated in figure 8, when the number of nodes is high, SPT is rather inefficient even at area close to sink due to path diversity. Therefore, we speculate that performance of this approximation algorithm degrades as node number increases.

[27] studied the energy efficiency of aggregation tree to some extent. But the main focus of this paper is to propose an approximation of SMT called Greedy Incremental Tree(GIT) and study its performance, therefore the scope of this paper is different from this paper.

In its simulation study, [27] compared the energy dissipation of GITs and SPTs. Since the energy model they use is different with ours, their results is not directly comparable to ours. Nonetheless, this paper pointed out that SPT and GIT are similar in low density networks but achieve significant energy savings at higher node densities (each node has more neighbors). This observation is similar to the observation we made in this paper.

3.6.2 Related Works on Data Aggregation Tree Efficiency

[13] first systematically studied data-centric routing approaches in wireless sensor networks. However, the focus of [13] is on comparing data-centric routing with traditional end-to-end routing scheme(address-centric routing). In this paper, address centric routing scheme is defined as shortest path tree without aggregation (overlapped paths are counted separately). Therefore, the emphasis of [13] is to compare the performance differences between "aggregate" and "do not aggregate", while our work investigates energy cost differences between aggregation aware and unaware schemes (SPT with aggregation and DB-SMT tree). So the focus of the two works are different.

[38] compares two major classes of data aggregation scheme: routing-driven compression (RDC) and compression-driven routing (CDR) across a broad range of spatial correlations. In this paper, RDC routes data through shortest paths toward sink, and performs opportunistic aggregations when routes overlap with each other. For CDR, routes are selected in order to compress data from all sources sequentially. This work mainly investigates the impact of correlation degrees on optimal aggregation structure. While for our study, we consider not only correlation degrees, but delay bounds and other parameters when comparing efficiency of correlation aware and unaware data aggregation trees. [38] uses grid topologies to compare RDC and CDR performance for different correlations, and we use more general uniform distribution topologies. Therefore, in this paper, the cost ratio of RDC over CDR is higher than the bound we observed in our study. We suspect that this difference is caused by the grid topology used in their paper. Nonetheless, results from this paper indicate that CDR outperforms RDC for high correlation scenarios, whilst CDR performs better for low correlation scenarios, which is comparable to our conclusion in more

general network settings.

3.7 Conclusions

In this preliminary research we study the energy efficiency of correlation aware aggregation trees in wireless sensor networks. Sensor applications with and without delay tolerance are considered, and how delay tolerance and other network conditions affect the efficiency of a correlation aware aggregation tree is explored. Through quantitative study and analysis, we conclude three rather surprising results: the energy improvement in using correlation aware aggregation is not significant under many network scenarios when the cost and complexity incurred in the tree construction process are taken into account; the maximum useable delay bound required to achieve the best possible energy efficiency is not high compared to the delay along the maximum length shortest-path in the default shortest path tree; and SMT is the optimal aggregation structure only when the correlation degree is relatively high. Practical implications of these results are also discussed.

CHAPTER IV

SCALABLE CORRELATION AWARE AGGREGATION: THE SOLUTION

4.1 Research Contributions

From the first part of the preliminary research, we arrived at the conclusion that theoretically correlation aware data aggregation does not provide significant energy improvement under many circumstances, and the centralized coordinations usually required for its construction and maintenance may even offset the energy benefits.

However, from a practical perspective, the energy improvement of a correlation aware structure over that of a correlation unaware structure is still substantial (around 1.5 times energy improvement for the simulations). If we can design a correlation aware data aggregation structure that achieves close to ideal performance improvement and is simultaneously good for a wide range of sensor network configurations without incurring the overhead of centralized coordinations, the benefits of correlation aware data aggregation can be fully realized, and energy efficient data gathering can be achieved.

In the second part of the preliminary research, we present a simple, scalable, and distributed aggregation tree called *SCT* (Semantic/Spatial Correlation-aware Tree) that does not require any centralized coordination while still achieves potential cost benefits due to efficient aggregation. The SCT structure is instantaneously constructed during the course of a single query delivery and is a fixed structure that is efficient for wide range of node densities, source densities, source distributions and correlation degrees. The SCT approach, with its highly manageable structure, ensures low maintenance overhead of the aggregation structure, while also addressing the other practical challenges identified in Section 4.2.

The rest of the section is organized as follows: Section 4.2 identifies the different challenges in designing a practical, efficient correlation aware structure. Section 4.4 presents the SCT approach in detail. Section 4.5 evaluates the performance of SCT with ideal structures and practical implementation of shortest path tree (SPT). Section 4.6 presents the key design principles in the SCT approach and explains how it addresses the corresponding challenges. Finally Section 4.9 concludes this part.

4.2 Challenges

The main goal of this preliminary research is to design an efficient aggregation structure that minimizes the message complexity of data aggregation for a wide range of sensor network applications. In order to realize this goal, we identify the following important challenges and explicate the desirable properties of a solution that addresses the challenge.

4.2.1 Construction

The foremost consideration in building an aggregation structure is the manner in which the aggregation structure is constructed. We have already seen that existing approaches in WSNs are correlation unaware and approximate a Shortest Path Tree (SPT). In a SPT, aggregation is not efficient even if the data from the sources are highly correlated because the primary concern is to minimize the delay. The ideal solution for the aggregation structure would be a Steiner tree or a stochastic Steiner tree [29], which depends on whether complete source knowledge is available at the time of construction. However, both problems are NP-hard.

Even the approximation algorithms for constructing a Steiner tree impose requirements such as the information regarding the number of sources and their locations to be available at the sink *a priori*. This information could be obtained if the sink adopts a two-phase querying procedure, where the query is sent in the first phase and the responses collected reveal the location of the sources interested in responding to that query. In the second phase, the sink initiates the construction of the approximation of a Steiner tree to optimize the message cost of the data transmitted. However, the overhead involved is $O(k)$, where k is the number of sources. This overhead might be comparable or even exceed the message cost of the aggregated messages. Another drawback of such a two-phase approach is the delay incurred in determining required information.

Moreover, the types of queries and responses sent may influence the nature of the ideal solution required for aggregation. The queries and responses can be categorized as (1) single query and one-time responses from the sources, (2) single query and multiple continuous responses from the same set of sources and (3) single query and multiple responses from a varying set of sources. We will explain each of these classifications briefly and present the ideal solution for each classification.

In section 4.7 we discussed a two-phase qureying procedure for collecting information regarding the number of sources and their locations and then construct a near-optimal data aggregation tree. However, this solution is not desirable across all types of queries and responses as shown below:

- *One-shot queries and responses*: In this category, the sensors send a one-time query and the corresponding responses from the sources are also one-time responses. In this case, the two-phase procedure will be clearly infeasible both in terms of delay and message complexity, as the message complexity and delay of the fist phase is comparable to the second phase. In this case, if the probability distribution of sources is given, the stochastic Steiner tree [1] is the optimal solution.

- *Single queries and multiple responses from the same set of sources*: Here, the sink sends one-time queries but the responses from each sensor may be comprised of multiple packets. However, the set of sensors responding to the query remains the same over all the packets. While in this case, it may seem that the two-phase approach may provide low message complexities, the delay incurred in determining the number of sources could potentially limit its application. However, the network Steiner tree is still the optimal aggregation solution in terms of low message complexity since the number and locations of sources is known after the first packets from all sensors reach the sink.

- *Single query and multiple responses from a varying set of sources*: Here, the responses

[1] A stochastic version of the Steiner tree problem where the expected cost of the data aggregation tree is minimized.

to the one-time queries may comprise multiple messages but the responding source sets may vary with time. For this case, it is desirable to have a solution that is independent of the locations of sources or the number of sources. In this case, the optimal solution is neither a network Steiner tree nor a Stochastic Steiner tree. We define this problem as a generalized Stochastic Steiner tree problem.

In summary, a desirable practical solution should consider the tradeoffs between the overhead involved in the construction process itself on the one hand, and the message complexity of the aggregation structure on the other hand, and ensure that it is reasonably efficient across all query and response paradigms.

4.2.2 Maintenance

Once the structure is constructed, it may be required that the structure be modified or reconstructed after a certain period of time to accommodate load balancing, node failures or any other reasons. Ideally, the maintenance overhead of the structure should be negligible in terms of both message complexity and delay.

We will now consider two important reasons for reconstructing the aggregation structure and discuss the preferred characteristics for accommodating those considerations:

- *Load Balancing*: This is to ensure that the energy consumed by all the nodes is fairly even over a certain period of time. In any aggregation structure, aggregation nodes take the responsibility of receiving, compacting and transmitting aggregated messages, and hence on average consume more energy than non-aggregation nodes. It is therefore likely that these nodes fail much earlier than the other nodes, impacting the connectivity of the network. To address this issue, we would like to spread the role of aggregation node among all nodes so that the network does not become disconnected prematurely.

- *Node Failures*: The structure should also be resilient to node failures, which are common in sensor networks [28]. Otherwise, it is likely that some messages will never reach the sink even though there may be alternate paths available. Therefore, when node failure occurs, the structure should have the ability to adapt and form a different, near-efficient structure.

4.2.3 Synchronization Requirements

One of the main considerations for any aggregation scheme is the time each node has to wait before it aggregates the messages received from all sources downstream of it. We refer to these timing requirements necessary for aggregation as synchronization requirements. In the absence of such synchronization requirements, it is conceivable that messages from some downstream sources may arrive after aggregation at a particular aggregation node and hence need to be transmitted separately. This will increase the message complexity despite the existence of an efficient aggregation structure.

Ideally, a scheme should enable an aggregation node to wait until the arrival of messages from all sources downstream before aggregation is performed. One way to do this is by having a timer at every node and wait for the expiration of the timer based on a waiting function, similar to the one described in [62], before performing aggregation. In this approach, each node waits for a time corresponding to $MAX - i \times \Delta$ after the reception of the first response, where MAX is the maximum wait time proportional to the depth of the aggregation tree constructed, i is the hop count of the node from the sink and Δ is the average degree of a sensor node. When the timer expires, it is assumed that all the messages from sources downstream have arrived at this node and the messages are aggregated. One of the obvious problems of this approach is that the nodes use the average degree of the network as opposed to the degree of that particular node. However, it is difficult to set aggregation timers accurately, especially when the network topology changes. This timer inaccuracy may in turn cause imperfections in the aggregation. Also, the computational overhead in maintaining fine-grained timers makes it less desirable. On the other hand, if a coarse-grained timer is used, the possible aggregation cannot be maximized.

In order to address this problem, an ideal aggregation structure should facilitate event-driven aggregation and should rely on timing requirements only sparingly. In this case, the timers can be made coarse as they will not be used often.

4.2.4 Other Considerations

An aggregation approach should also be reasonably efficient in terms of message cost when the degree of correlation varies ($0 < \rho \leq 1$). In addition, it should be able to perform efficient aggregation irrespective of the distribution of sources. As we will see in Section 4.5, the proposed approach takes into account these considerations and performs reasonably well for a wide range of network scenarios.

4.3 The SCT Design Basis

The design of SCT is predicated on two important elements:

- An aggregation backbone facilitating the generation of efficient aggregation trees
- A fixed structure independent of source distribution and density.[2]

These two design elements address the challenge of efficient construction, and incorporate the characteristics and requirements of sensor networks. In this section, we establish and justify these two design elements. The details of SCT approach will be present in section 4.4.

4.3.1 Motivation for a Ring-and-sector Division

Our target problem of data collection with variable source distributions and densities can be thought of as a generalized version of the stochastic Steiner tree problem. The stochastic Steiner tree gives the optimal message cost when there is a given source probability while our goal is to find the efficient aggregation tree for variable source probabilities. In [24, 29], the authors have presented centralized constant-factor approximations to the stochastic Steiner tree problem. In this subsection, we will provide motivation of the structure we used to approximate the generalized stochastic Steiner tree, adopting similar arguments provided by the above two related works.

Consider the network model in which each sensor, i, has a probability, p_i, to report data to sink. If an edge e between two sensor nodes is used by a set of source nodes D to

[2] ratio of the number of sensors that send data packets to the sink to the total number of sensor nodes in the network

transmit data to sink, then the cost of this edge can be defined as:

$$c_e = Pr[e\ is\ active] = 1 - \prod_{i \in D}(1 - p_i) \tag{30}$$

This cost function captures the tradeoffs in the characteristics of data transmission and correlation in sensor networks as follows: when more sources use one particular edge, the total communication cost for this edge increases; on the other hand, the cost per message decreases due to the correlation between multiple messages. Since c_e is a concave function, if the number of sources using one edge is beyond a certain value, adding more sources causes only a minimal increase in the total cost. Therefore, it pays to place a certain number of aggregation nodes in the network with the property that all the edges between those aggregation nodes are highly utilized. The sources can then connect opportunistically to closest aggregation nodes using edges with a higher cost. With this structure, the total expected cost of the aggregation tree for a certain source probability distribution can be minimized.

Based on these observations, a good aggregation structure should have the following features: (i) A subset of nodes is chosen as aggregation nodes and a spanning tree is built on top of these nodes to form a "backbone" for aggregation; (ii) Each node in the backbone is responsible for aggregating messages from sources within a certain sub-area.

There are many ways to construct the aggregation backbone. In this paper, we design a uniform division of the network to assist in the construction of the backbone, based on the following criteria:

- The division of the network should facilitate the distributed selection of aggregation nodes within the network with no additional message overhead.
- The division should result in energy-efficient data aggregation irrespective of distribution of sources.
- The interconnection of aggregation nodes to form the backbone should be achieved in a distributed fashion with low overhead.

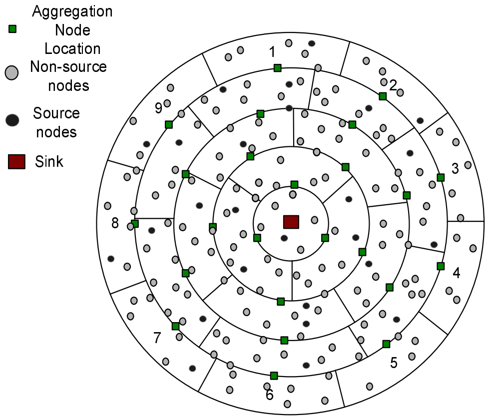


Figure 9: SCT Structure: Rings, Sectors and Aggregation Nodes

- The division of the network should ensure easy connectivity of all non-backbone nodes to the aggregation backbone.

In SCT, one such efficient division of the network is adopted, where the network is divided into rings and sectors. The division not only addresses *all of the above* desired criteria, but is also characterized by several impressive practical benefits, including *lack of global synchronization requirement* for aggregation of sensor-data, *ability to perform load balancing* with no additional message overhead, *ability to address node failures*, etc. *The ring-sector division is also characterized by the unique invariant that a message sent by any source always propagates towards the sink at each hop, because of the symmetrical structure of the division.* We elaborate more on the motivations of ring-sector division in SCT later in this section and in Section 4.4.

As illustrated in Fig. 9, the network is divided into m concentric rings with the same width (R/m). Each ring is in in turn divided into sectors of the same size such that on average each sector contains about n_0 nodes. For each sector, an aggregation node is chosen as a member of the aggregation backbone, and each aggregation node in ith ring is connected to it's upstream aggregation node in $(i-1)th$ ring via shortest path. The collection of all aggregation nodes and shortest paths form the backbone aggregation tree. Each aggregation node is responsible for collecting messages from all sources in the sector it belongs to.

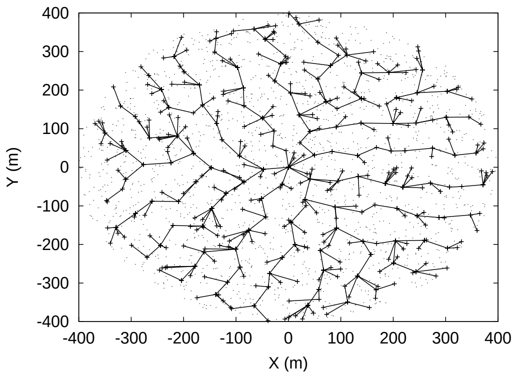
As we will see in the Section 4.4, this structure can facilitate the realization of a desirable aggregation backbone in a distributed fashion. But the problem is only partially addressed because our goal is an optimal aggregation structure for variable source densities. Therefore, the next question is what is the optimal number of the aggregation nodes. An immediate observation is that with increasing number of sources, the number of optimal backbone aggregation nodes increases. This can be explained intuitively as follows: when the number of sources is small, probability of aggregating two or more messages from different sources at an aggregation node is relatively small. Moreover, having more aggregation nodes translates to a backbone with higher cost, because of the additional transmissions required by the aggregation nodes. Therefore, fewer aggregation nodes are more desirable.

However, as source number increases, the probability that two or more messages from sources being aggregated at each aggregation node increases. Therefore, addition of aggregation nodes helps in aggregating the messages from different sources as early as possible. In this case, the additional cost incurred by introducing extra aggregation nodes can be offset by the reduced high-transmission cost edges used due to early aggregations. Hence, as source density increases, more aggregation nodes are desirable.

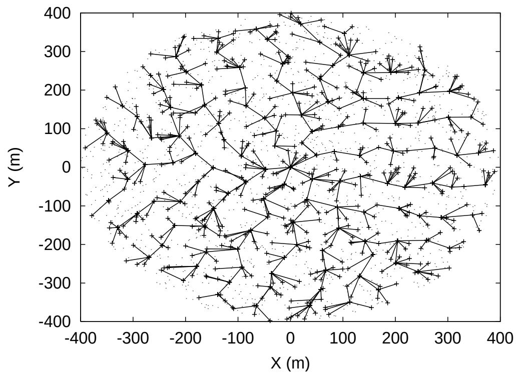
As we identified above, a good approximation of optimal structure should adapt with different source densities: the higher the source density, the more nodes are involved in the backbone. This indicates that the proposed ring-sector structure should also adapt to source densities in order to approximate the optimal solution. However, in the next subsection, we will show that a fixed structure satisfying certain properties is reasonably efficient for a wide range of source densities, and hence motivate a relatively stable aggregation structure with low maintenance overhead.

4.3.2 Motivation for a Source-independent Aggregation Structure

In the previous sub-section we pointed out that for the ring-sector structure proposed, the optimal number of aggregation nodes increases with the source density. In this sub-section, we will show that when the source density increases beyond a certain point, the optimal structure stays the same because of a “saturation” phenomenon. Thus, by choosing a fixed aggregation structure, we are able to do efficient aggregation for a large range of source densities while incurring very little construction overhead.



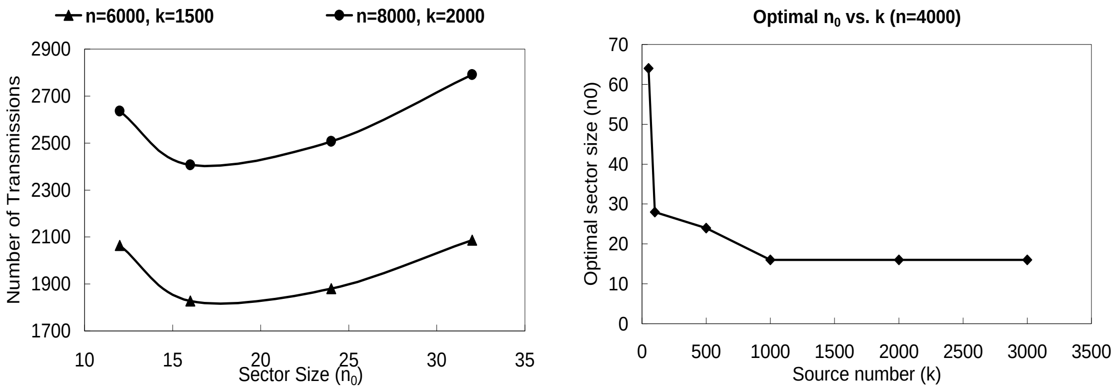
(a) $k = 250$



(b) $k = 500$

Figure 10: The Optimal Structures When $n = 2000$

The optimal sector size generally reduces with increasing source density. But this reduction is not always desirable. Consider a certain threshold source number k_0 at which the optimal sector size is small enough such that aggregation node just falls into transmission range of every other node in the sector. We call the size of the sectors at this point the *saturation size*. In this case, every source message can reach aggregation node with 1-hop transmission. Reducing sector beyond the saturation size will not help increasing aggregation efficiency. Furthermore, the introduction of additional aggregation nodes increases the message cost. Thus, when the number of sources, k, is increased beyond k_0, decreasing the sector size results in increased message cost. Therefore, when $k > k_0$, the optimal aggregation structure should remain the same. Figures 10 (a) and (b) are visualizations of the aggregation trees when each sector reaches saturation size. These figures show the SCT aggregation tree constructed over all source nodes and aggregation nodes. The scattered points on the background are non-source, non-aggregation nodes. Figure 10 (a) is the optimal aggregation tree when $n = 2000$ and $k = 250$, and figure 10 (b) is the optimal structure when $n = 2000$, and $k = 500$. From the two figures we can observe that the "backbones" of the aggregation trees are the same, only the sources increase in the latter case. This is because saturation is already achieved when $k = 250$. Therefore, even if k increases, the optimal structure remains the same. Notice that the above "saturation" phenomenon is not specific to the ring-sector structure we proposed, but is true for the optimal aggregation structure in general due to the fixed transmission ranges of all sensor nodes.



(a) Message cost variation (b) Optimal sector size variation

Figure 11: Find the Optimal Sector Size

The effect of saturation on message cost is also substantiated by other simulation results. Figure 11 (a) shows for a certain n and k how the message cost of aggregation tree varies with varying sector size n_0. When n_0 is small, the message costs of the aggregation trees decrease as n_0 increase. However, after $n_0 = 16$, costs increase with increasing n_0, indicating saturation for both cases.

Figure 11 (b) shows how the optimal n_0 varies with k/n. For a wide range of source densities, the optimal structure is the same. This implies that a fixed aggregation structure is good enough most of the time in our application. Simulation results for other node densities give similar results.

We now describe how the optimal values for m[3] and n_0[4] can be estimated at the sink. Both parameters are chosen such that each sector in the network reaches the saturation size. The choice of m and n_0 is explained next.

The choice of m determines the depth of the aggregation tree. Since the downstream aggregation nodes are sources for the aggregation nodes one ring closer to the sink, to achieve saturation size, level-i aggregation nodes should be within 1-hop distance to level-i+1 aggregation nodes. Therefore, m can be determined by the following formula:

$$m^* = \frac{R}{r}\beta \tag{31}$$

where R is the radius of the network, r is the transmission range of the nodes, and β is

[3] the total number of rings in network, or the aggregation levels
[4] the average number of sensor nodes in each sector, determines the sector size

a constant used to accommodate the fact that aggregation nodes are not distrubted on a straight line. We choose a empirical value of 1.32 for β based on experimental results.

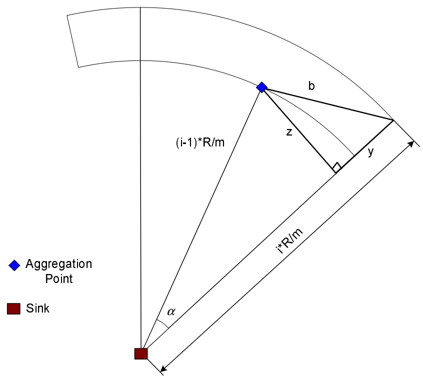


Figure 12: Maximum Travel Distance within a Sector

Determination of n_0 is based on the requirement that every node within this sector is less than 1-hop away from the aggregation node. Refering to figure 12, the furthermost possible node within this sector is located at the corner of this sector, we indicate the distance of this node to aggregation point as b. The distance b is calculated in the right-angled triangle as follows:

$$z = \frac{(i-1)R\sin\alpha}{m} \tag{32}$$

$$y = \frac{iR}{m} - \frac{(i-1)R\cos\alpha}{m} \tag{33}$$

$$b = \sqrt{z^2+y^2} \tag{34}$$

$$= \frac{R}{m}\sqrt{(4i(i-1)\sin^2\frac{\alpha}{2})+1} \tag{35}$$

The angle α for the ith ring is given by:

$$\alpha = \frac{m^2 n_0 \pi}{2(2i-1)n} \tag{36}$$

When $\alpha \to 0$, $sin\frac{\alpha}{2} \to \frac{\alpha}{2}$, and equation 35 reduces to

$$b = \frac{R}{m}\sqrt{\frac{1}{2}(\frac{m^2 n_0 \pi}{2n})^2+1} \tag{37}$$

To make sure b is less than transmission range, we let $b = \sigma r$, where $\sigma < 1$ is a constant. From equations 37 and 31, we can derive the optimal n_0^* for large k as:

Table 1: Comparison of Computed and Experimental Parameters

Node Density	Computed m^*	Experimental m^*	Computed n_0^*	Experimental n_0^*
n=2000	8.54	9	13.9	16
n=4000	11.6	12	14.9	16
n=6000	14.1	15	15.5	16
n=8000	16.1	17	16.0	16

$$n_0^* = 0.428(\frac{r}{R})^2\sqrt{\sigma\beta^2 - 1} \tag{38}$$

Through empirical studies, we determine $\sigma = 0.836$. To verify the accuracy of the choice of constants, we compare optimal m and n_0 derived from equation 31 and 38 and those obtained from simulations. Table 1 shows that the theoretical values match experimental optimums closely.

4.4 The SCT Approach

In this section, we present an overview of the SCT approach and explain the event-driven aggregation in detail.

4.4.1 SCT Structure Overview

In this section, we explain the SCT approach in detail. We present the different phases of the data aggregation process as well as provide insights for the design choices.

4.4.2 Division of the Network

During the setup phase, the sink propagates the following information to the entire network: (i) location of itself, (X_s, Y_s), (ii) the total number of nodes, n (iii) the radius of the network, R and (iv) the computed values for m and n_0 to all the nodes in the network. Each node in the network is assumed to know it's own geographical location. When a node receives the packet, it first computes the distance between itself and the sink. This determines the ring, i, to which it belongs to. For example, any node at a distance d from the sink, such that $(i-1)\frac{R}{m} < d \leq i\frac{R}{m}$, belongs to the ith ring. Each node can calculate the number of sectors per ring $s(i)$ as: $s(i) = \lceil\frac{2(i-1)n}{n_0 m^2}\rceil$. Given the locations of the node with respect to the sink and the number of sectors within the ring, any node can determine the sector number to which it belongs.

4.4.3 Determination of Aggregation Nodes

Once the network has been divided into sectors and rings, the aggregation node for each sector has to be selected. In the SCT approach, the aggregation nodes are selected by leveraging the fixed, geometric division of the sensor field.

For each sector in a given ring, the geometric center of the lower arc bounding the sector is defined as the ideal location of the aggregation node for this sector. The node closest to this ideal location is chosen as the aggregation node. Given the value of m and n_0, each source not only knows the sector and ring numbers to which it belongs but can also determine the boundaries of the sector. If we are to adopt polar coordinates and if α and β are the bounding angles of a sector corresponding to the ith ring, the location of the aggregation node is given by $((i-1)\frac{R}{m}, \frac{\alpha+\beta}{2})$.

Each source within this sector routes messages to this location of the aggregation point. During the query forwarding phase, each node piggybacks the coordinates of itself along with the query and other information sent by the sink. In this fashion, a node can learn the locations of its immediate neighbors during the query delivery phase. When a source wants to send it's message to the ideal location of the aggregation point, it sends the packet to the node closest to the ideal location. Once a node receives this packet, it then does a local broadcast declaring itself as the aggregation node. Since most of the nodes within any sector are in a one-hop region of the aggregation node, the other source nodes will then forward their packets to this aggregation node. In this way, the node closest to this ideal location becomes the aggregation node where messages are aggregated. In a few cases, it is possible that two sources may elect two different nodes as aggregation nodes simultaneously. However, such instances are rare and do not increase the message cost considerably because of the following two reasons: (i) most of the nodes in a sector are within a 1-hop region of each other, and (ii) once a node is elected as the aggregation node, the local broadcast will enable other sources to identify this node as the aggregation node.

Thereafter, these aggregation nodes act as sources for the sectors in the next ring closer to the sink, and send aggregated messages to aggregation nodes of those sectors. With this approach messages are combined step by step as they progress towards the sink, until the last ring is reached, for which the sink is the aggregation node. Figure 13 is a illustration of this aggregation process.

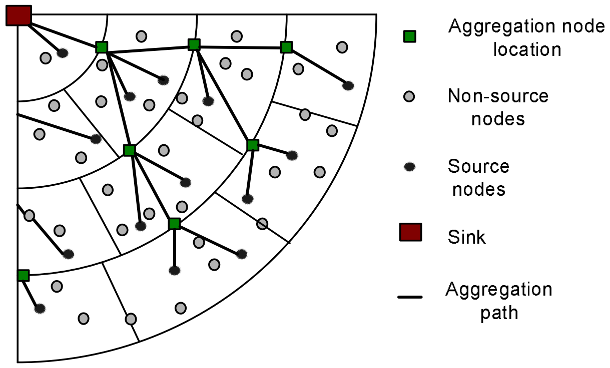


Figure 13: A Subsection of the SCT Aggregation Structure

The aggregation nodes are implicitly elected by the routing protocol when the sources first send their data to the location of the aggregation nodes. To ensure that an aggregation node is elected at every sector irrespective of the presence or absence of sources in that sector, we adopt the following procedure in the last ring: (i) During the query forwarding phase, some nodes in the last ring identify themselves as corner nodes within each sector. The corner nodes are located at the periphery of the upper arc bounding a sector and farthest from the ideal location of the aggregation node. (ii) These corner nodes take on the responsibility to identify the physical aggregation node for this sector by forwarding a dummy packet to the geographical location of the aggregation node. Note that this procedure needs to be done only for sectors in the last ring as these aggregation nodes act as sources when communicating to the upstream aggregation nodes.

This geometric election of the aggregation nodes facilitates independent and distributed computation of aggregation point locations by each node. It also enables a location based routing approach to be used to send packets from the sources to the aggregation nodes or between aggregation nodes. The geometric structure construction and aggregation nodes election allow each node to have a consistent view of the entire network to help set up the

SCT structure in a distributed fashion.

4.4.4 Event-driven Data Collection

To achieve maximum aggregation of the source data at the aggregation nodes, it is also necessary to ensure that these nodes wait for an optimum delay value. In Section 4.2, we identified the drawbacks of using a fine-grained aggregation timer to trigger the aggregation process. In SCT we use a more desirable alternative where there are only coarse-grained timers and the aggregation process is mainly event-driven. This approach is motivated by the fact that each aggregation node knows the exact number of children that are also aggregation nodes. When an aggregation node receives information from all children that are aggregation nodes, it is assured that the data from all sources within the sector are also received by an aggregation node. This is because the sources transmit their data at the beginning of each message collection round while the aggregation nodes wait for the notifications from all the downstream aggregation nodes. The arrival of messages from all downstream aggregation nodes is used as the trigger to merge and propagate the information collected, upstream towards the sink. Thus, synchronization is achieved in an event-driven fashion without the need for explicit delay timers at each aggregation node.

The aggregation node identification procedure in the last ring also helps in determining the time to aggregate and forward messages to the upstream aggregation node. Even if there are no sources within a sector, the aggregation node sends out a small message to the upstream sector indicating the absence of sources. The arrival of messages from all the downstream aggregation nodes is used to trigger the aggregation process in that aggregation node. In this way, the aggregation process is mainly *event driven*. For the case when there are no nodes in a downstream sector, there will not be any downstream aggregation node in that sector. In these rare cases, the aggregation nodes use the coarse-grained aggregation timer before aggregating the messages from other downstream aggregation nodes and sources. Note that the probability of occurrence of such empty sectors is extremely low as we will discuss in Section 4.7.

4.4.5 Load Balancing

Load balancing schemes, as we have identified in Section 4.2, are important to ensure that the resources of all non-source nodes are utilized to roughly the same extent over a period of time. We propose two simple schemes to distribute the roles of aggregation nodes to different sets of nodes over a certain period of time:

1. *Location of the rings*: In the current SCT description, the different rings are of width $\frac{R}{m}$, where R is the radius of the network and m is number of rings. For load balancing, the location of the first ring can be shifted by a distance $\frac{R}{m} - rc$, where r is the one-hop transmission range and c is a small integer that is varied from $0 \ldots \frac{R}{mr}$. The same offset is applied to every ring so that the width of the ring is still maintained to be $\frac{R}{m}$ for all rings except the first and last.

2. *Orientation of the sectors*: In a similar way, we can choose the offset angle for a sector to be different across multiple queries. The offset angle, θ, can be incremented according to the relation, $\theta = \frac{c}{s(i)}$ where $s(i)$ is the number of sectors in the ith ring and c is a small integer dependent on the query identifier. This again assures that different nodes are chosen as aggregation nodes over several query floods.

4.4.6 Aggregation Reliability and Node Mobility

The failure of any non-aggregation node will not impact the correctness or efficiency of the SCT approach. Therefore, here we only discuss how the aggregation node failures are addressed in SCT. Recall that we require an aggregation node to announce itself to its neighbors once it receives the first message from a source. In this way, aggregation node failures before the setup of the SCT structure can be identified by the lack of announcement from the particular node, and another node close to the ideal location can announce itself as the aggregation node of this sector. If an aggregation node fails during the information collection phase, the lack of ACK messages from this node to sources can inform them of its failure. In this case, the retransmission of the first packet enables the election of a new aggregation node, which in turn broadcasts an announcement upon receiving the retransmitted message. After the re-election, aggregation proceeds as usual. Notice that the election of a new aggregation node may delay the entire aggregation process due to the retransmissions and announcement necessary, but the correctness and efficiency of the aggregation process will not change since it is event-driven.

In the case of node mobility, the proposed solution needs to be modified to accommodate mobility in (i) sources, (ii) aggregation nodes and (iii) other nodes. If the sources are mobile but the aggregation nodes are fixed, sources will forward to the closest aggregation node given its current location. If aggregation nodes are mobile, the node failure handling mechanism can be leveraged to elect a new aggregation node for that sector. Mobility of other nodes does not affect the SCT approach.

4.5 Performance Evaluation

In this section we evaluate the performance of the SCT approach under different network configurations and compare it with two centralized schemes: minimum Steiner Tree, SPT, and one decentralized scheme: DSPT (Decentralized Shortest Path Tree). We vary the node density, source density, source distribution, as well as correlation coefficient (ρ) and evaluate the message cost of the four structures under different scenarios.

4.5.1 Simulation Environment

- We use a discrete event simulator based on the LECS simulator for all evaluations. The simulation topologies are largely similar to those used in general sensor networks: 2000 to 8000 nodes are uniformly distributed within a circular field of radius 400m. The number of sources that generate messages for one specific query varies as $\frac{1}{10}$, $\frac{1}{6}$, $\frac{1}{4}$ and $\frac{1}{2}$ of the total number of nodes in the network.

- We evaluate the SCT approach using two metrics: message cost and data gathering latency. For message cost, we measure the total number of transmissions required for all responses to reach the sink for one round of data collection, and for data gathering latency, we measure the time interval from the time when all sources start to send messages, till the last message reaches the sink.

- To focus on the comparison of aggregation efficiency of different structures, we assume a perfect MAC layer that avoid collisions of data packets.

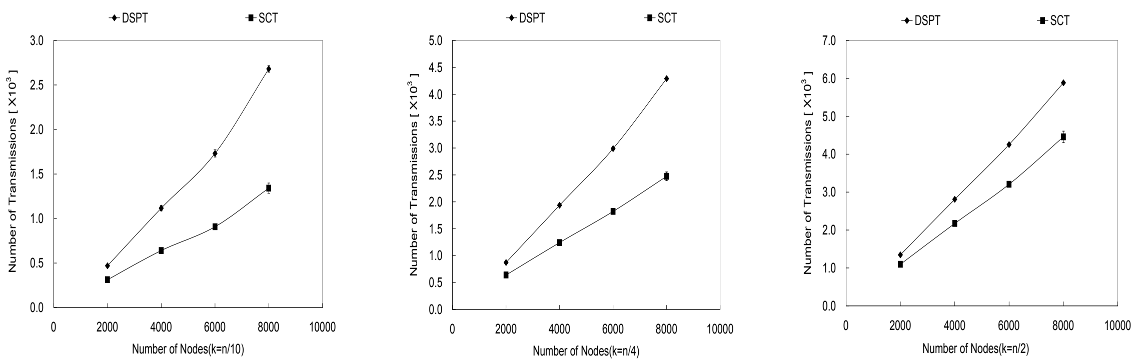
- All the simulation results are derived after averaging results over 10 random seeds and are presented within 95% confidence intervals.

- Since minimum Steiner tree is the optimal solution when sources are fixed, we compare SCT with an approximation of minimum Steiner tree (MST) generated using Prim's algorithm. We also compare the SCT with SPT generated by Dijkstra's algorithm because it is representative of correlation-unaware structures. To highlight the benefit of SCT as a distributed solution, a decentralized version of the shortest path tree (DSPT) is also included in the evaluation. In DSPT, GPSR routing protocol is used to approximate SPT in a distributed fashion because the routes generated by GPSR closely approximate SPT, especially when node density is high [30].

4.5.2 Perfect Correlation ($\rho = 1$) Scenarios

4.5.2.1 Different Node Densities

We first compare the performance of the decentralized SCT with that of the DSPT. In this scenario, we assume that data from all sources are correlated perfectly ($\rho = 1$). We will address the ($\rho < 1$) case later in this section.

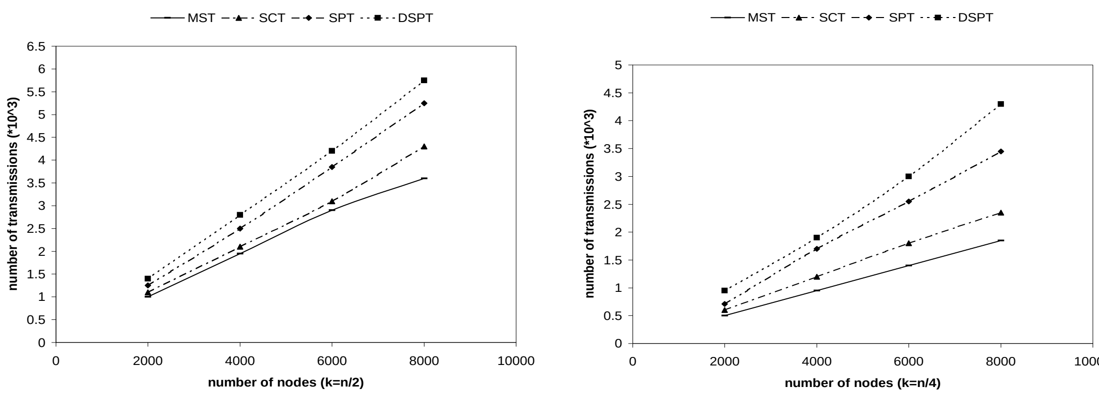


(a) $k = n/10$ (b) $k = n/4$ (c) $k = n/2$

Figure 14: Performance Comparison between SCT, and DSPT: Figure (a), (b), and (c) Show the Number of Transmissions as a Function of Number of Nodes for Different Source Densities k

Figures 14 (a), (b), and (c) show the cost of two decentralized schemes as a function of the number of nodes for different numbers of sources k. In these simulations, we choose the total number of nodes n as 2000, 4000, 6000, and 8000, and the number of sources k as $\frac{n}{10}$, $\frac{n}{4}$, and $\frac{n}{2}$, respectively. To ensure fair comparison, we assume DSPT uses an explicit mechanism to achieve perfect aggregation, therefore the message cost is a measure of the aggregation structure efficiency only (We will present results related to delay performance of both schemes later). It can be seen that SCT outperforms DSPT scheme under all situations. Interestingly, we observe that the cost of DSPT is up to 200% of the SCT cost as the number of nodes increases. The DSPT cost also increases faster than the SCT cost as node number increases. This is expected since more nodes reduce the efficiency of aggregation in DSPT as the paths chosen by different sources are less likely to overlap. Therefore, SCT can be considered a more scalable approach. Furthermore, it is observed that the difference between the two schemes increases as the ratio of the number of sources to the number of nodes, $\frac{k}{n}$, decreases because more sources increase the probability of aggregation for DSPT. As the ratio $\frac{k}{n}$ approaches 1, both schemes converge into n transmissions.

4.5.2.2 Decentralized vs. Centralized Schemes



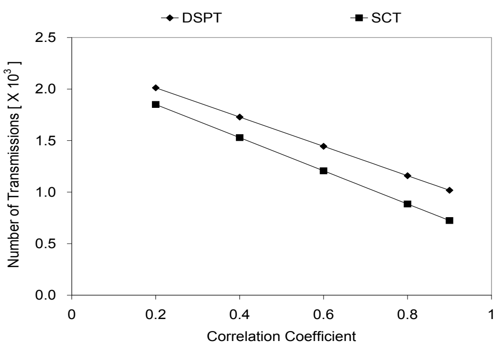
(a) $k = n/4$ (b) $k = n/2$

Figure 15: Performance Comparison between SCT and Centralized Schemes, SPT and MST: Figure (a) and (b) show the Number of Transmissions as a Function of Number of Nodes for Different Number of Sources k

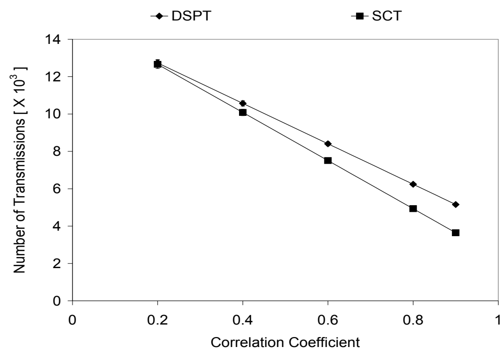
We compare the performance of the decentralized SCT and DSPT schemes with the centralized schemes they approximate. Figure 15 shows the cost of the proposed scheme and the centralized schemes as a function of the node number. To evaluate the cost of the centralized schemes, we assume perfect aggregation for both SPT and MST. From the figures we can see that DSPT's message cost approaches closely that of SPT while SCT's message cost approaches closely the cost of MST. Furthermore, although SCT is a decentralized scheme without perfect aggregation, it still outperforms the centralized SPT, since SCT explicitly aggregates sensor data, while SPT just leverages aggregation opportunistically. We also observe that as the k/n ratio increases, the difference between both decentralized schemes and their approximated centralized schemes decreases, because as k increases, both schemes can achieve better aggregation and approach the performance of an ideal structure.

4.5.3 Different Correlations $(0 < \rho < 1)$

In the extreme case of $\rho = 1$, minimum Steiner tree is the optimal aggregation structure, while for $\rho = 0$ (no correlation), SPT is the optimal aggregation structure. For the general case $0 < \rho < 1$, no optimal solution exists. And the problem is classified as NP-complete in [16].



(a) $n = 2000, k = 500$



(b) $n = 8000, k = 2000$

Figure 16: Performance Comparison between SCT and DSPT in Case of Different Correlation Factor ρ

In Figure 16, we characterize the message complexity of SCT when the correlation coefficient varies from 0.2 to 0.9. Notice that in this graph, the number of transmissions is normalized to a unit message size. For example, if after aggregation, a node transmits a message of size 1.5 times the unit message size, it is counted as 1.5 transmissions. From this figure, we can see that for both DSPT and SCT, message cost reduces as ρ increases, since the two schemes have either implicit or explicit mechanisms to leverage the correlation. However, the message cost of SCT reduces faster than DSPT because it facilitates aggregation at an earlier stage of packet forwarding, hence can reduce packet transmission cost more effectively. Since DSPT is the optimal structure when $\rho = 0$, we expect that the two curves will cross each other at a certain small correlation factor. For $n = 8000$, the cross point occurs at $\rho = 0.2$, while for $n = 2000$, the cross point is expected to occur at $\rho < 0.2$. As we analyzed earlier, DSPT aggregates less efficiently when n becomes large, hence it does not lose much when ρ decreases, therefore it performs relatively better for a smaller ρ when n is large.

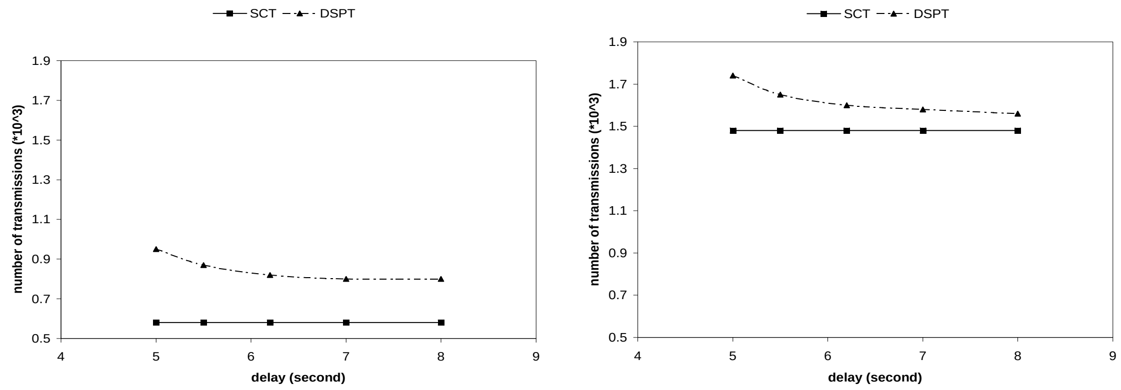
4.5.4 Delay Sensitivity

To evaluate a data gathering scheme, latency is another important metric besides the message cost. In this part, we study the data gathering latency of SCT and at the same time characterize the message cost-latency trade off for SPT.

Since SCT is an event-driven approach, none of the aggregation nodes on the tree needs a timer, and each aggregation node can forward messages as soon as it receives responses from downstream aggregation nodes. Therefore, there is very little overhead for perfect aggregation. While for SPT, in order to achieve perfect aggregation, each node on the tree has to set a timer and wait for a certain amount of time before aggregation.

Since SPT itself does not include any mechanism for aggregation timing, we implement a simple scheme [62] that sets aggregation timer for each node based on its hop distance to the sink. Each node on the tree knows its distance (in hop count) as well as the maximum distance D among all nodes on the tree to the sink (depth of the tree). Given a maximum delay MAX_DELAY, per-hop delay Δ is calculated as MAX_DELAY/D. A node that is i hops away from the sink will wait for $MAX\_DELAY - i * \Delta$ time before aggregating and forwarding data it has received. In this way, if Δ is large enough to accommodate transmission and contention delay at each hop, perfect aggregation can be achieved.

Figure 17 shows the number of transmissions as a function of maximum delay MAX_DELAY for $n = 2000$ and $n = 4000$ cases. It is shown that SPT achieves perfect aggregation



(a) $n = 2000$ (b) $n = 4000$

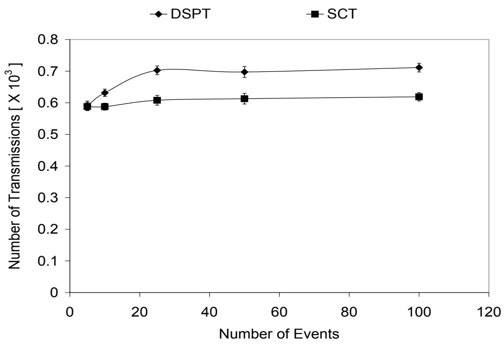
Figure 17: Performance Comparison between SCT and SPT in Case of Different Delays

when MAX_DELAY is more than 8.0 second. However, the cost of SPT increases as MAX_DELAY approaches 5.0 second since smaller MAX_DELAY increases the possibility of late arrivals of data before aggregation. Once a packet misses its aggregation deadline at one of the intermediate hops, the probability of it missing deadlines at later hops becomes higher, which explains why SPT performance aggravates quickly as MAX_DELAY decreases. On the other hand, since SCT is an event driven data aggregation structure, its message cost remains the same irrespective of different delays, which explains the flat curves in both figures.

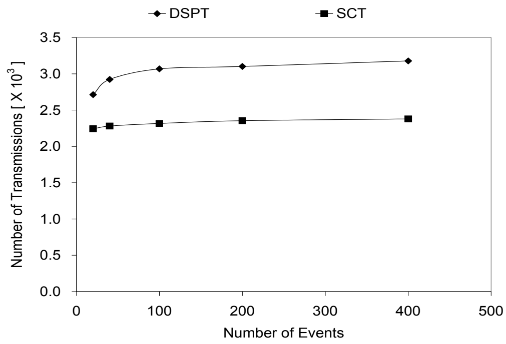
4.5.5 Localized Events

Until now, we have assumed that sources are deployed independently in a sensor field. This is typical when considering semantic correlation, where each source reports information about a discrete event. However, spatial correlation usually generates non-uniformly distributed sources, where multiple sources are reporting the same event. In this case, sources are clustered around one or more spots in the network. In this section, we investigate how SCT performs when sources are distributed in such a fashion.

In the simulation, we consider $n = 2000$ and $n = 8000$ scenarios, each has a total of $k = \frac{n}{4}$ sources in the network. The number of events (spots) in the network is varied from $\frac{k}{100}$, $\frac{k}{50}$... to $\frac{k}{5}$. With this configuration, the total number of sources does not change, but the distribution varies from a highly concentrated scenario to close to uniform distribution.



(a) $n = 2000$



(b) $n = 8000$

Figure 18: Performance Comparison between SCT and DSPT for Different Numbers of Localized Events in Case of 2,000 and 8,000 Nodes, Respectively

Figure 18 (a) and (b) show the simulation results for such spot topologies. When there are very few events and source distributions are highly concentrated, DSPT achieves good aggregation at early stage and aggregated messages reach the sink along only a few paths. But SCT also performs well because it does explicit aggregation within each sector that contains sources, while other sectors without sources or not on the aggregation paths to the sink do not take part in the aggregation process, therefore does not incur any extra cost. From the figure, we can see that at $n = 2000$, when the number of events is small, DSPT and SCT have similar cost. But as the number of events increases, sources are less concentrated as in the previous case, and the probability of DSPT paths overlapping reduces, hence the higher message cost. However, for SCT the only extra cost is the introduction of more aggregation nodes into the aggregation tree, which is limited by the total number of sectors in the network. Therefore, the increase of message cost is much less compared with DSPT. For $n = 8000$ case we observe a similar trend, and even for the least event number, SCT outperforms DSPT, because as we analyzed earlier, the chance of DSPT paths overlapping decreases as node density increases.

4.6 Analysis of SCT Structure

The above performance analysis shows that SCT provides good energy performance across a wide range of node density, source density, source distribution and correlation degrees. This implies that the same SCT structure can be used even when the sensor network configuration changes. For example, as we discussed Section 4.2, there are different query-response types. For most of them, the aggregation structure needs to be constructed beforehand, and should be efficient across different source densities and distributions, which is true for SCT structures. Furthermore, if the correlation model of information collected by the sensors changes, the same SCT structure is still good for a large range of correlation degrees. Therefore, SCT is a desirable and practical data aggregation solution for wireless sensor networks.

In this section, we explain the underlying principles of SCT design. We also relate the performance results and structure characteristics of SCT to the observations and inferences obtained from the first part of the preliminary research, and analyze the reasons for SCT to achieve good performance for a wide range of sensor network configurations.

4.6.1 The Adaptability of SCT to Node Density Variations

From the performance comparison, we concluded that similar to centralized schemes such as SMT and MST, SCT achieves significant energy savings over SPT for a wide range of node densities.

Recall that in the previous analysis, we mentioned that SPT becomes less energy efficient as node density increases since the path diversity of SPT increases. A centralized scheme can reduce the diversity by explicitly combining paths that are close to each other when node density is high. However, SCT achieves this path merging in a distributed manner.

When constructing a SCT backbone, the network is divided into sectors such that each sector is about the same size and contains on average n_0^* nodes. Since n_0^* is a fixed number, it depends only on r and R. As node density increases, the sector size reduces. Because each source sends message directly to the aggregation node in the same sector, and downstream aggregation nodes send aggregated messages to aggregation nodes one level up directly, the possible path diversity is reduced as much as possible, therefore, SCT performs reasonably well under a wide range of node densities. Of course, the introduction of aggregation nodes may result in higher message complexity, which explains the less efficient performance of

SCT compared with centralized correlation aware solutions, the low construction and maintenance overhead of SCT as a distributed approach may result in better net performance than those centralized schemes.

4.6.2 The Adaptability of SCT to Source Density and Distribution Variations

As mentioned in Section 4.2, the optimal aggregation structure from sources with varying source distributions and densities is a generalized version of stochastic Steiner tree. In [24, 29], the authors presented centralized constant-factor approximations to the stochastic Steiner tree problem. In this subsection, we use similar arguments to motivate SCT as an approximation of the generalized stochastic Steiner tree.

Given the aggregation backbone structure, the next question is what is the optimal number of the aggregation nodes. For SCT to be energy efficient, the backbone structure should adapt to the source density variations. In another word, the sector size of SCT should decrease as source density increases. However, we have shown in the 4.3 section that when the source density is beyond a certain value, the optimal structure no longer changes because of a "saturation" phenomenon.

As a result, in SCT design, the sector size is chosen based on the condition that all the nodes in one sector can reach the aggregation node of this sector with one-hop transmission, and this structure is efficient for various source densities.

4.6.3 The Adaptability of SCT to Correlation Degree

From part I of our preliminary research, we arrived at the conclusion that SMT is a good aggregation structure only when the correlation degree is close to one. For low and medium correlation degrees, DB-SMT the same maximum delay as that of SPT performs much better in terms of energy efficiency. As we explained earlier, under these circumstances, it is advantageous to deliver each message along the shortest route possible. At the same time this makes full utilization of en-route aggregation possible.

The structure of SCT closely matches this criteria since the number of rings on SCT is the network diameter in number of hops. Hence, the maximum path length of SCT is close to that of SPT. Since each source on the SCT transmits directly to its corresponding aggregation node, the aggregation paths are close to shortest paths in hop length. At the same time, as we discussed earlier, SCT explicitly facilitates path aggregation by defining an aggregation backbone and reduces path diversity to a great extent. Combining these two factors, the energy performance of SCT resembles that of DB-SMT for a wide range of correlation degrees, and therefore is a good solution for general sensor network applications.

4.7 Issues and Discussions

4.7.1 Empty Sectors

In Section 4.4, we assumed that each sector contains at least one node which can serve as an aggregation point and ensure that the tree is connected. However, due to the irregularity of Poisson node distribution, it is possible that some of the sectors may not contain any node. Here, we investigate the implications of having an empty sector. In this case, downstream aggregation nodes of the empty sector won't find a path to aggregate, and hence will not be able to forward the aggregated information. The design of SCT has a simple back up strategy in this case: if an aggregation node is not able to find an aggregation node upstream, it will simply forward the aggregation message to a neighbor in the adjacent sector. After that, the aggregated message will be delivered via a connected aggregation path.

Although this is a less efficient aggregation path, we argue that the probability of empty sectors is extremely low for the target environment.

Consider a network with n nodes and S total number of sectors of the same size, the probability that no empty sector exists in the network is:

$$\begin{aligned} P &= \frac{S^n - \sum_{i=1}^{S-1} \binom{s}{i} (S-i)^n (-1)^{i+1}}{S^n} \\ &= 1 - \sum_{i=1}^{S-1} \binom{s}{i} (1 - \frac{i}{S})^n (-1)^{i+1} \end{aligned}$$

where

$$S = \sum_{i=1}^{m} s(i) = \sum_{i=1}^{m} \lceil \frac{(2i-1)n}{mn_0} \rceil \tag{39}$$

Table 2: The Probability of Non-empty Sectors

Node Density	Probability
n=2000	99.97%
n=4000	99.99%
n=6000	99.94%
n=8000	99.89%

For the n_0^* and m^* we chose for the fixed structure, the probabilities of non-empty sectors for various node densities are shown in Table 2.

4.7.2 Comparison with Clustering Approaches

Structure-wise, SCT approach bears a few similarities with clustering algorithms proposed for ad-hoc and sensor networks [11], [60]. In a clustering approach, the entire network is partitioned into clusters of similar size, with a head node elected in each cluster. Non-cluster nodes report messages via a short path to their corresponding cluster heads, where messages are aggregated and transmitted to the sink, and the total transmission cost is reduced through the decreased total transmission distance and smaller message size. With this analogy, a natural question to ask is: *How is SCT different from these clustering algorithms?*

In terms of cluster construction, there are two different solutions from existing works. One approach is for each node to broadcast in a certain area its properties (id, node degree, residual energy etc.), following which an election process is executed to choose the cluster head [60], [10]. This approach generally assures regular cluster size and full node coverage, but at the cost of high communication overhead. Using the approach proposed in [60], the clustering algorithm is triggered at periodic intervals to select new cluster heads. At each interval, the clustering process requires a certain number of iterations to finally select the desired cluster head. If the minimum probability of a node becoming a cluster head is p, it takes $N \leq \lceil \log_2 \frac{1}{p} \rceil + 1$ steps for the election algorithm to terminate ($N = 6 - 15$ iterations for average scenarios). During each iteration, a tentative cluster head generates broadcasting messages, resulting in a total message cost (for setting up the cluster structure) to be of order

$N * n$, where n is the total number of nodes in the network. This translates to significant energy consumption, given that the election process is invoked repeatedly to achieve load balancing. In contrast SCT does not require the overhead of message exchange in either the initial set up phase or in the later maintenance phase, where structure modifications are performed.

Another approach, such as the one used in [11], is to specify a certain probability for each node to become a cluster head, and the node which turns out to be cluster head announces itself through a limited-scope flooding. This approach incurs lower message overhead, but cannot guarantee a uniform cluster head distribution and full coverage of all non-cluster nodes. Therefore, the *orphan* nodes not covered by any cluster heads have to transmit their messages directly to the sink, significantly increasing the total message cost. On the contrary, in SCT, aggregation nodes are chosen implicitly without any message exchange, since the location of the node is the criteria for head selection. Therefore, in terms of clustering forming overhead, it is comparable to the probability-based approach. On the other hand, the clustering structure formed in SCT is organized such that each node is guaranteed to be covered by an aggregation node, and every cluster has similar size. In this sense, a low message cost is achieved without incurring any overhead for additional message exchanges.

In terms of cluster manageability, SCT benefits from its highly regular structure. Since cluster heads consume more energy in terms of communication and computation, cluster head rotation is a essential part of any clustering scheme. For most of the clustering schemes, rotation involves re-election of cluster heads, which incurs a lot of overhead. While with SCT, since the structure is symmetric along all radii, a simple rotation of the SCT rings and sectors moves the roles of cluster heads to a different set of nodes in the network. With carefully designed rotation sequence, the role of cluster heads can be evenly distributed to all sensors in the network.

4.7.3 Impact of Network Shape and Sink Location

In the previous sections, we assume that the sensor network has a circular shape and the sink is located at the center of the network. These assumptions are made for ease of discussion and presentation of SCT. The approach itself does not impose any constraint on the shape of the network and the location of the sink. For example, consider a network with rectangular shape and the sink located at one of the four corners of the rectangle. In this case, the SCT structure can still be constructed by fitting the entire network into a polar coordinate, with sink being the root of the SCT and the diagonal of the rectangle being the radius. Load balancing can be achieved as well by shifting the rings and sectors periodically as described before. Although the efficiency of the SCT structure for irregular network shapes may not be as high as that of circular network shape, the correctness of SCT is not compromised.

4.8 Related Work

4.8.1 Correlation Unaware Approaches

We now consider two of the popular choices used in the design of routing protocols in sensor networks: (i) using the query paths to construct the sensors-to-sink routes and (ii) using the location information of sensors and the sink to forward messages to the sink. Most of the contemporary routing approaches [28, 30, 41] fall under one of the two categories in terms of basic routing design principle. We consider directed diffusion [28] and GPSR [30] as representative examples of (i) and (ii), respectively, and show that the structure generated by these approaches can be translated to a Shortest Path Tree (SPT).

In directed diffusion [28], sink requests data by sending interests for named data. The interests are propagated through the network wherein every node delivers the interest to all its neighbors through a local broadcast. During the diffusion process, all nodes set up gradients towards their neighbors from which the interest was received. While there are different possible criteria for the reinforcement of the gradient, the most common practice results in a shortest path tree rooted at the sink [16, 17].

GPSR is a geographic routing protocol that eliminates the need to maintain states while performing routing. In this approach, every source in the sensor field knows the geographical location of the sink, and addresses the message for sink with the specific location. For each hop along the path to sink, a node chooses the node closest in geometric distance to the sink as the next hop destination and forwards the message to it. Since GPSR delivers a vast majority of packets in the optimal number of hops [30], the data gathering tree again approximates shortest path tree.

From the previous discussion we establish that two of the representative routing schemes in sensor networks are approximations of shortest path trees. Since the primary goal of this structure is to minimize delay, shortest path tree is not a correlation-aware data gathering structure. Even though opportunistic aggregation may still occur when different paths overlap with each other, this structure does *not* maximize the aggregations possible in the network. A correlation-aware structure, on the other-hand, would be able to cut down message cost by explicitly facilitating data aggregation in the network.

4.8.2 Correlation-Aware Approaches

Several related work have been proposed in the context of explicit aggregation [16,17,24,29]. We categorize them into two general classes: correlation-aware structures assuming complete global source knowledge and incomplete source knowledge.

4.8.2.1 Structures Built with Complete Source Knowledge

When the full knowledge about source location is known, the Steiner Tree over all sources, sink and non-source nodes gives the optimal message cost when the degree of correlation is close to 1. However, the computation of Steiner Tree is an NP-hard problem [49].

In [16, 17], the authors propose simple heuristics that approximate the Steiner tree to perform efficient aggregation when messages are correlated. For any given correlation factor $0 < \rho \leq 1$, the authors describe (i) leaves deletion heuristic and (ii) balanced SPT/multiple Traveling Salesman Problem (TSP) tree as simple alternatives of the Steiner tree.

In [8], the authors first identify that the message cost can be modeled as a concave-cost function for any correlation factor $0 < \rho < 1$, and propose an algorithm that constructs approximation trees simultaneously good for all concave cost functions.

However, these approaches require complete information regarding the number of sources and their location to be available at the sink and cannot work for the cases when the information is incomplete. While such information can be made available via queries and responses, the overheads involved in acquiring such information both in terms of message cost and delay could be potentially prohibitive. Moreover, they are centralized approaches hence do not scale well with increasing node densities typical to WSN environments. Finally, these approaches try to solve the problem of efficient aggregation from a theoretical perspective and do not consider the practical challenges that we identify in Section 4.2.

4.8.2.2 Structures Built with Incomplete Source Knowledge

[29] and [24] address the more general problem of building aggregation structure with optimal expected cost when the knowledge of sources is incomplete.

The problem considered by both work is a network with a root node and a collection of N client nodes in the network. Each client, i, may choose to contact the root independently from others with some probability p_i along a path from itself to the root. If a client chooses to contact the root, the edges on this path become active. The goal is to minimize the expected number (or cost) of active network edges over a random choice of the clients. This problem is a stochastic version of the deterministic Steiner tree problem.

The stochastic Steiner tree problem is an NP-complete problem. Therefore, the focus of this work is on developing constant-factor approximation algorithms. In [29], the authors observe that the optimum solution is invariably a tree, and the optimum tree consists of a central "hub" area within which all edges are almost certainly used, together with a fringe of "spokes" in which multiple clients contribute independently to the cost of the solution. To set up a good approximation structure, their solution leverages a facility location algorithm to identify a good set of "hubs" to which clients route messages at independent costs, and the set of hubs are connected using a Steiner tree algorithm. In [24], the authors design a similar structure constructed in two stages: during the first stage, a subset of nodes D is chosen by picking each node independently from the network with probability proportional to p_i, then a minimum spanning tree is built over D; later revealed clients requesting services are connected to the existing structure with an augmentation algorithm. Using the above principle, the authors propose a 2-approximate algorithm for the stochastic Steiner tree problem.

Both papers propose good constant-factor approximation algorithms. However, they are not distributed solutions and are not tailored for sensor network environments. For example, [29] requires building a Steiner tree over "hubs", and [24] requires building a minimum spanning tree to form a first-stage backbone. Both need centralized computation with high complexity. Also, they assume a fixed probability for nodes responding to a query and do not handle the situation where the source density is variable. Finally, they do not address all the practical challenges we identified in Section 4.2.

4.9 Conclusions

In this research, we propose a novel solution to aggregate correlated information from a subset of sensors to the sink. The proposed scheme is scalable, distributed, requires minimal computation and is highly-manageable compared with existing solutions. The proposed scheme is assessed both intuitively and analytically. Through simulations, we compared the proposed scheme with ideal, centralized data structures and a distributed structure. Simulation results show that as a correlation-aware structure, SCT performs significantly better than correlation-unaware structures in terms of message cost and data gathering latency.

CHAPTER V

BACKGROUND

In the previous part of the proposed research, we addressed the energy-efficient data gathering problem on a macroscopic level. Specifically, we evaluate the performance benefits of correlation aware data gathering structures and the overhead involved in using them. We also proposed a distributed and scalable correlation aware data gathering structure to realize the benefits with relatively low overhead.

In this part, we will address the energy-efficient data gathering problem on a different level. The single hop communication that forms the basis of data gathering process in sensor networks are investigated, and a novel communication strategy is proposed which enables the transmission of every message in the data gathering process with much lower energy than what is normally required. This strategy can reduce the energy consumption of data gathering operations significantly in WSNs.

The problem of minimizing the energy dissipation for transmitting a set of packets over a single-hop link has been studied in several research works, [7,39,46,52,59,61]. All of them are based on a physical layer technique called *modulation scaling* [46].

Modulation scaling achieves efficient wireless communication by adapting the modulation levels (bits per symbol)of data to be transmitted. For example, in Quadrature Amplitude Modulation (QAM) as an example, one symbol in QAM can represent 2 bits, 4 bits, 64 bits, etc, depending on the constellation size selected for modulation. As the number of bits per symbol decreases, the energy consumed for transmitting the same amount of information can be reduced for the same bit error rate (BER), due to less transmission power required to achieve the same signal to noise ratio (SNR). However, the delay involved in delivering the same amount of information increases, because less bits of information are contained in each symbol. Therefore, *modulation scaling essentially reduces the energy dissipation during communication at the cost of increased latency.*

This technique has been widely explored in the field of sensor networks since it helps address the primary concern of energy conservation. Most of the works propose optimal packet scheduling algorithms to minimize energy consumption for given latency constraints. [39] investigate the problem of minimizing the energy dissipation for transmitting a set of packets over a single-hop link subject to a specified latency constraint. In [7], the problem is extended to consider a single transmitter and multiple receivers. [46] propose an on-line policy for adjusting modulation level for single-hop communication, and in [52], modulation scaling is integrated into the Weighted Fair Queuing (WFQ) scheduling policy. [61] extends the problem to a multi-hop communication environment, and finally [59] addresses packet scheduling in a general tree structure.

The advantage of modulation scaling is the flexibility it provides in optimizing the energy consumption by exploiting the energy-delay tradeoff. But it also has its own disadvantages: (i) The range of adaptation is limited; not all modulation schemes have this option, and for schemes where modulation scaling is possible, the minimum and maximum bits per symbol allowed are usually within a small range, depending on the nature of the modulation. (ii) The granularity of adaptation is not infinitesimal; the number of bits per symbol only takes integer values, and typically can be varied only as powers of two. (iii) The extended transmission time of data packets when applying lower modulation levels degrades channel utilization for shared wireless medium.

Given this context, we ask the following question: is there other technique that provides similar power control flexibility like the modulation scaling without the limitations and drawbacks we identified above? Our answer is in the affirmative and we will explore a new communication strategy that exploits energy-delay tradeoff of data transmission in our proposed research.

CHAPTER VI

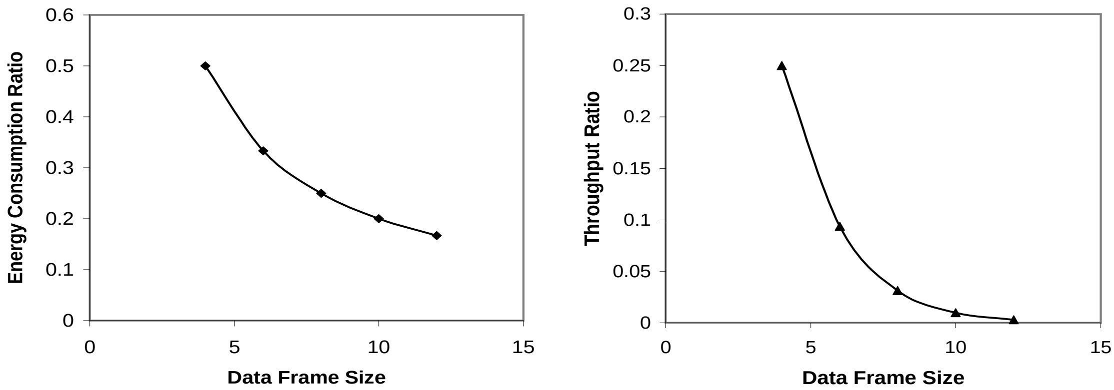
COMMUNICATION THROUGH SILENCE: THE CONCEPT

6.1 CtS Fundamentals

Most if not all of the current communication applications assume a common mode of communication that we refer to as *Energy based Transmissions* (EbT). It involves the use of energy driven bit transmissions for sending information from a sender to a receiver. For example, when a sensor s_1 wants to communicate a *value* 97 to a neighboring sensor s_2, it sends the sequence of bits (1, 1, 0, 0, 0, 0, 1) in succession using a certain amount of energy for every bit transmitted. Thus, if the energy consumed per bit transmitted is e_b, the total transmission energy consumption is $7 * e_b$.

In the proposed research, we explore a new communication strategy that is based on conveying information using silence as opposed to energy based transmissions. In the above example, s_1 would send a *start* signal to s_2, which would then start *counting* up from zero. s_1 knowing the rate at which s_2 is counting, would send a *stop* signal when it knows s_2 would have counted up to the value 97. When s_2 receives the stop signal, it stops counting and treats the value in the counter as the information transmitted by s_1. If the start and stop signals can each be sent with the same energy e_b, the total energy consumed is $2 * e_b$, which is better by a factor of 3.5 when compared to the EbT scheme. We refer to such an approach that uses silence for communication as *Communication through Silence* (CtS). Note that we ignore issues of headers, synchronization, etc., in the above discussion for clarity, and just consider the net information to be sent by the sender to the receiver.

While it is evident that the CtS Strategy can deliver considerable amounts of energy improvement, there is a key trade-off involved in the form of throughput reduction. Considering the same example as before, while the value 97 was transmitted in 7 bit slots when using EbT, the same value will take 97 clock cycles when using CtS. If the clock rate of



(a) Ratio of bits transmitted (b) Throughput Ratio

Figure 19: The Energy-Throughput Tradeoff in Basic CtS (CtS:EbT)

the sensor is of the same order as the data rate of its radio, this translates into an exponential decrease in the throughput enjoyed by the sensor. Secondly, even if the throughput problem is overlooked (say, for delay insensitive applications) or addressed through appropriate optimization mechanisms, there still remain several challenges related to how traditional medium access control (MAC) functionalities such as framing, addressing, sequencing, error-control, and contention resolution can be performed when using a strategy such as CtS.

In this context, we made three contributions in this research:

- We introduce the new paradigm of *Communication through Silence* for wireless sensor networks, define the concept of CtS, and identify its basic trade-offs.
- We present unique optimization strategies that can be employed while using CtS that either alleviate the serious throughput trade-off, or in some cases even improve upon the throughput performance as compared to the traditional EbT scheme.
- We identify and discuss several research challenges that exist in realizing both the optimization strategies and traditional MAC layer functionalities.

The rest of the paper is organized as follows: Section 6.2 introduces CtS, Section 6.3 and 6.4 discuss the potential applications of CtS. Section 6.5 describes several optimization strategies that can be employed with CtS in order to improve its throughput performance while retaining its energy benefits. Section 6.6 identifies several challenges in realizing a

MAC protocol for the CtS approach. Section 6.7 discusses related work, and finally Section 6.8 summarizes this section.

6.2 Overview

The conventional communication strategy - EbT - involves the use of energy driven bit transmissions for sending information from a sender to a receiver. Generally, if the information to be sent by the sender at any given instance spans k bits, the energy consumption is $k * e_b$. Note that we ignore issues of headers, synchronization, etc., for clarity. In this context, we introduce a new communication strategy that uses *silent periods* to convey information from the sender to the receiver. We refer to such a strategy as *communication through silence* (CtS). The start and stop signals in the example presented in section 6.1 can be assumed to be one-bit transmissions. The energy consumption for CtS is then always $2 * e_b$, irrespective of the value being sent. Note that the energy consumption for counting per clock cycle is considerably smaller than the energy consumption for transmission, and more importantly the counting clock (or the system clock), which has to be active even for the conventional communication, can be tapped into, thus not incurring any additional overheads. Not surprisingly, the above performance improvements will grow as the magnitude of the value being transmitted increases. For example, if the value being conveyed spans 20 bits, the savings grow to a factor of 10 or about 900%. Essentially, with increase in the magnitude of information being conveyed, the energy consumption of CtS stays at $2 * e_b$, while that of *EbT* keeps increasing.

The trade-off, however, lies in the delay needed to convey any piece of information. Assuming that the clock rate is the same as the data rate of the underlying communication channel, the CtS strategy, as described earlier, incurs exponential increase in delay when compared to EbT. Thus, *the throughput of the basic CtS strategy is substantially lower than that of EbT.* Figure 19(a) and (b) shows the energy (number of bits transmitted) and throughput ratio between CtS and EbT scheme. Since each frame takes only two signals (start and stop) to transmit, the energy consumed for transmit each bit of data decreases inverse linearly with frame size. However, the throughput of transmitting each frame decreases exponentially with the number of bits in the frame, as opposed to EbT, which always take 1 bit slot to transmit 1 bit. Essentially, the throughput of CtS decreases as $\frac{s}{2^s}$, where s is the frame size.

While we discuss later in the paper how the performance of the basic CtS strategy can be augmented to not only alleviate the decrease in throughput, but in many cases go beyond that of EbT while preserving the same benefits in energy consumption, we now provide a formal definition of the *basic* CtS strategy:

To deliver a binary packet of size k and form $n_{k-1} \ldots n_1 n_0$ (where $n_i = 0, 1$ for all $i = 1, 2 \ldots k-1$), sender interprets the bit stream as a value N as follows:

$$N = 2^{k-1} \times n_{k-1} + 2^{k-2} \times n_{k-2} + \ldots + 2 \times n_1 + n_0 \tag{40}$$

and transmits N using only two signals: a start signal and a stop signal, with the time between the two signals being the time taken by the receiver to count up to the value N from zero. The receiver, knowing N and k (which is the standard frame length in bits), infers the bit stream $n_{k-1} \ldots n_1 n_0$.

6.3 Potential CtS Applications

Since the throughput and energy consumption of CtS scheme is usually lower than those of EbT (however, we will show later that under some network configurations the throughput of CtS can be *higher* than that of EbT), the niche applications of CtS are low-power and low-rate communications between small or portable wireless devices. A wide range of existing applications fall into this category. Among them are sensor networks, wireless personal area networks, wireless body area networks etc. The common characteristics of these networks are the limited power supply of network devices, and the relatively small amount of information exchanged for a given time interval. For example, in sensor networks used for environment monitoring, wildlife tracking and industrial automation, sensor data is collected on an hourly or even daily basis, therefore the networks are usually not heavily loaded. Sensor networks are not the only type of networks with these characteristics. Consider low-rate wireless personal area networks (LR-WPAN) [5], which interconnect small portable wireless devices such as RFIDs, PDAs, special purpose sensors and actuators(such as those for home automation). For those applications, the message exchange between those devices are limited, usually contains only inventory information, sensor readings or management information. A body area network contains independent nodes such as sensors distributed in the clothes or even implanted into human body. It is an indispensable element of wearable computing [51] and has rich applications in home/health care, sports, defense, ambient intelligence, pervasive computing and many other areas. For Body area networks, energy efficiency is a major concern due to the limited power supply of sensor devices. Also, it is desirable to reduce wireless transmissions as much as possible to reduce the impact of radiation on human health. At the same time, the throughput requirement of these networks is typically not high. Therefore, it's possible to tradeoff throughput for energy efficiency. CtS enables applications to consume minimum energy on data transmission while at the same time maintain a reasonable data throughput, hence is a proper communication strategy for those applications. However, communication protocols designed for conventional communication strategy cannot make use of the potential of CtS properly, therefore, the practical application of CtS calls for new hardware and communication solutions designed specifically for this new strategy.

6.4 CtS and Wireless Sensor Networks

While the above description of CtS was for a generic wireless link, the focus of our research is the study of how CtS can be employed in the specific context of wireless sensor networks (WSNs). Certain unique characteristics of WSNs enable the consideration of CtS in a better light. Firstly, one of the fundamental challenges faced when using CtS is the throughput decrease that comes along with the energy improvements. WSNs represent an environment where energy consumption can be of prime importance with the amount of energy left at nodes directly determining the survivability of the network. Hence, delay tolerance from an application's standpoint might be higher especially when it can lead to considerable energy savings. Secondly, certain physical characteristics of WSNs, such as all sensors eventually wanting to communicate with the sink, sensor devices primarily having homogeneous computing and communication capabilities, etc., are suitable for an effective realization of CtS.

There are two key goals that need to be addressed in order to be able to use CtS as a practical communication solution in WSNs:

- *Improving the energy-throughput trade-off:* In the basic CtS strategy described earlier, for an improvement order of α in energy consumption, the throughput reduction incurred is of the order of 2^{α}. Thus, consider a WSN application in the network that can operate at 10Mbps of raw data rate. Using CtS with a CtS data frame size of 10 bits, the effective data rate is reduced to 10Kbps ($=\frac{10^7}{2^{10}}$) for an energy improvement of a factor of 5 ($=\frac{10}{2}$). While it is true that certain applications might trade-off throughput for energy performance improvement, the scope of applicability or use of CtS in WSNs is likely to be increased tremendously if the energy-throughput trade-off can be alleviated or even improved. Hence, we ask this question: *Can the throughput performance of CtS be improved using additional techniques such that the throughput reduction is lesser or absent while retaining the energy improvements?*

- *Practical realization:* The energy-throughput trade-off is a fundamental theoretical and potentially an algorithmic problem to solve. However, another important challenge to be addressed is how CtS is practically realized in a WSN setting. Questions such as what type of radio is appropriate, how large the CtS data frame sizes need to be, how addresses are conveyed, how sequencing is performed, how errors are detected, how contention resolution is done, etc., have to be tackled in order to practically realize the potential improvements CtS can provide. We thereby ask this question: *What is an appropriate physical layer configuration and medium access control (MAC) strategy for CtS, with the latter tackling issues such as framing, addressing, sequencing, error control, and contention resolution?*

Given these goals, our next step is to design optimization strategies to improve the throughput properties of CtS without compromising its energy benefits, as well as come up with MAC layer and system solutions that enable its application in WSNs.

6.5 Optimization Strategies

In this section, we explore communication strategies that are possible due to the unique characteristics of CtS, and can improve upon the considerably low throughput performance of basic CtS.

The strategies retain most of the energy benefits of the basic CtS scheme, while improving upon the throughput. All strategies presented still use start and stop signals as before. The strategies differ from the basic scheme in how they use the time between the start and the stop signals. Essentially, if the start and stop signals are addressable (which they have to be for practicality), the intermediate *silence* between them can be used for other purposes to increase the throughput. The three strategies are complementary in nature, and can be used in tandem. Note that the strategies presented in this section do not form a comprehensive optimization set when using CtS. Instead, the goal of the section is to introduce a few possibilities in communication enabled using CtS.

Similarly, we use a few preliminary results obtained through simple simulations to illuminate desirable properties of the discussed optimization strategies. The simulation results hence do not represent an exhaustive evaluation of the optimization strategies, which although desirable is outside the scope of the thesis. In the rest of the section, we first present the three strategies, along with illustrative performance results that compare the CtS performance with and without the strategies. Finally, we show how CtS with the three strategies combined performs with respect to EbT.

6.5.1 Evaluation Model

For the rest of the discussion in this section, we make the following assumptions: (i) the start and stop signals are uniquely addressable and occupy a limited number of bits, but for simplicity we will assume one bit transmission slot each for the time being, (ii) the communication channel is lossless, (iii) the sender and the receiver clocks are perfectly synchronized, and there are no counting errors, (iv) the route from a sensor to the sink is made available by an independent routing protocol, and (v) the sequence of CtS frames (with values) to be transmitted are available in the buffer (i.e. the framing is already performed). We make the above assumptions to simplify the discussion of the strategies. We refer to the segment of information CtS sends using a pair of start-stop signals as a *CtS frame*, and its content as a *value*.

All results presented are derived from a simple custom built event-driven simulator that allows for packet transmissions and receptions in pre-configured sensor network topologies. The simulator does not incorporate elements such as channel losses, jitter etc. However, collisions and retransmissions are recorded, and appropriately accounted for in the performance evaluations. We discuss the specific network topologies used, and any other assumptions made as we present the results. The energy results are based on a model that incorporates transmit, receive, idle, and sleep energy values of 1100mw, 900mw, 100mw, and 20mw respectively based on [3]. The measure for energy consumed is energy/bit, and the measure for throughput is defined as bits/bit slot which indicates how many information bits are actually transmitted in one transmission clock cycle. 95% confidence intervals are presented for all data points.

6.5.2 Multiplexing

Consider two contending links L_{12} and L_{34} between sensors s_1 and s_2, and s_3 and s_4 respectively. In EbT, when L_{12} has an ongoing transmission, L_{34} has to remain idle. For example, if s_1 has to send information, say the value 8, to s_2, while s_3 also has to send information, say the value 13, to s_4, L_{12} will have to first complete its transmission of the value 8, before L_{34} can perform its transmission of the value 13. Hence, information transmission has to be inherently sequentialized as long as the links in question are contending links.

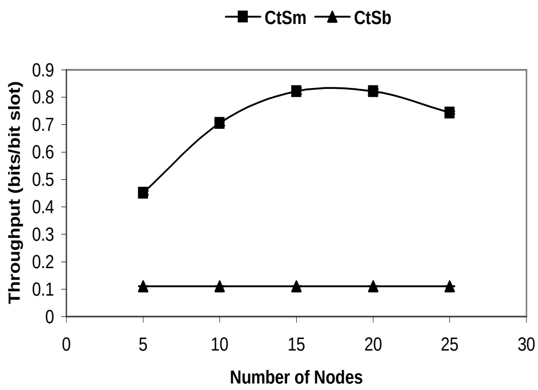
However, while using CtS, even if the links are contending, such information transfer can be performed in *parallel*, by multiplexing the start and stop signals appropriately. Using the same example as above, if s_1 sends the start signal in bit time slot t_i, the stop signal has to be sent in bit time slot t_{i+8}. However, s_3 can send its start signal in bit time slot t_{i+1}, which in turn will require it to send its stop signal in slot t_{i+14}. Thus, while in the basic CtS approach, the two transmissions would have taken a total of 21 slots, the multiplexing has allowed the two transmissions to be completed in 14 time slots.

We refer to the ability to send multiple overlapping CtS frames simultaneously as *multiplexing*, and CtS with multiplexing as CtS_m in the rest of the paper. Note that such multiplexing can be done irrespective of whether the contending links in question share vertices or not. Specifically, if in the above example s_2 and s_4 are the same sensor, as long as the start and stop signals are uniquely identifiable (with both source and destination addresses), the sensor will still be able to receive the multiplexed signals. The same applies when the source vertex is shared by the two links. While multiplexing can be performed on the same link for multiple CtS-frames, such multiplexing can be done in a far more efficient fashion in the further discussion. Also, such multiplexing cannot be performed using EbT, as otherwise the frames will arrive corrupted at the respective receivers if the links are contending. Multiplexing in CtS can thus be formally defined as follows:

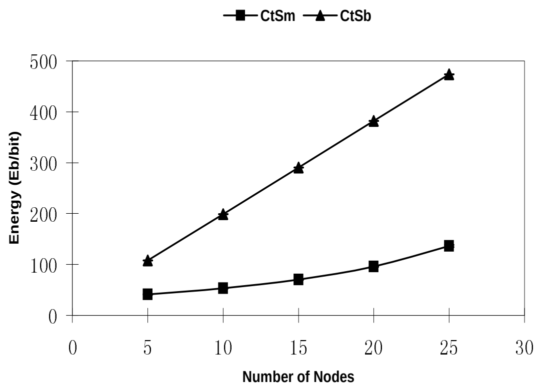
Multiplexing: *If a link L has a scheduled transmission of a CtS frame with start and stop signals in bit time slots t_i and t_{i+k}, any other CtS frame on a contending link can be scheduled as long as the start and stop signals of the new frame are not transmitted in slots t_i and t_{i+k}.*

How the multiplexing is controlled by the sensors in a distributed fashion is an important problem. However, we defer discussion about the problem till Section 6.6, and assume a simple ALOHA like scheme for the rest of the section. Essentially, a sensor transmits its signals independent of the other sensors, and multiplexing occurs naturally. Figures 20(a) and (b) show the simulation results of the throughput and energy consumption performance of the basic CtS approach (CtS_b) and CtS with multiplexing (CtS_m) using the simple ALOHA like scheme respectively. The results are shown for varying number of contending sensors in a *one-hop region*. In this configuration, the transmission of each sensor is capable of reaching all the other sensors in the region. Note that multiplexing is essentially a strategy for communicating over a single link.

Figure 20(a) shows the throughput performance, and it can be observed that CtS_m achieves about a $4x$ improvement in throughput as compared to CtS_b. The primary reason for the improvement is the ability to multiplex multiple CtS frames over a given time window. The throughput performance of CtS starts to decrease when the number of nodes



(a) Throughput Performance



(b) Energy Performance

Figure 20: Impact of Multiplexing

in the contention region is more than 15. This is because of the increase in the collisions and retransmissions of start/stop signals. In practice, this problem can be alleviated by adapting the CtS data frame size to reduce contention. However, in simulations we keep the same frame size for consistency.

The energy performance in Figure 20(b) is presented in terms of the average unit energy consumed per information bit transferred. It can be observed that the energy performance of CtS_m is better when compared to CtS_b, and the difference increases as the total number of nodes in the contention region increases. This is due to the less time spent by nodes in the idle state. When multiplexing of CtS frames is possible, the network throughput increases, hence the total amount of time used to transmit all the CtS frames is less, and the proportion of time nodes spend in the idle state is less. While using CtS_b, each node spends considerable amount of time waiting for start/stop signals from other nodes since multiplexing is not allowed. Therefore, the energy consumption of CtS_b is much higher, and this difference is magnified by the number of nodes in the contention region. Although in CtS_m, more collisions and retransmissions are possible, which can cause each node to transmit more start/stop signals than in the case of CtS_b, the resulting extra energy consumption is compensated by the previous factor. This shows that the multiplexing strategy can not only improve the throughput performance of CtS_b, but also save on energy.

While the results presented serve as a simple proof of concept for the efficacy of the CtS_m scheme, the problem of performing a more intelligent distributed scheduling of CtS frames to leverage multiplexing is an important one. We revisit the problem again in Section 6.6.

6.5.3 Cascading

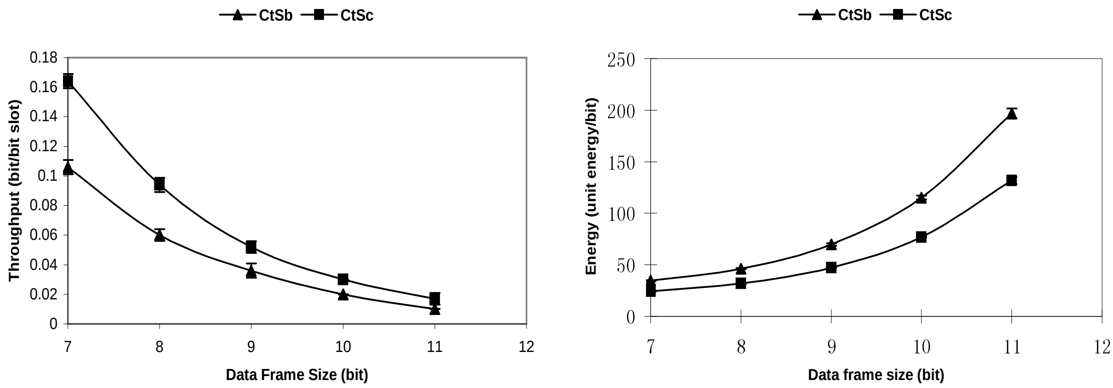
Consider sensor s_1 having to send CtS frames with values of (7, 13, 19, 28, 10, 6, 8, 21) to a neighboring sensor s_2. In CtS_b, the above information can be sent in a total of 112 bit time slots (sum of the values). Assuming 5 bit EbT data frames, this translates into a throughput utilization of 35.71% ($\frac{40}{112}$). However, assume that CtS is augmented with an *intermediate* signal in addition to the basic start and stop signals. When a receiver receives an intermediate signal, it records the value it has currently counted up to, but *continues* counting for the next value instead of resetting the counter to zero. In the above scenario, s_1 can then send a start signal at time slot t_i, intermediate signals at time slots t_{i+7}, t_{i+13}, and t_{i+19}, and a stop signal at time slot t_{i+28}. Note that it has to "stop" at 28 since the next value to be transferred(10) is less than 28. Thus, four values can be transferred in just 28 bit time slots for an effective throughput of 71.43% (the overall throughput for the entire sequence using the same methodology is 67.8%). Furthermore, the number of signals transmitted is now reduced to 5 from the original 8 (4 start/stop signals) required in CtS_b.

We refer to this ability to build on previous transmissions to send subsequent transmissions as *cascading*. We refer to CtS with cascading as CtS_c. Again, if the start/stop/intermediate signals are uniquely addressable, the cascading can be done not just between the same pair of sensors, but also between different pairs of sensors with the intermediate signals carrying the address of the specific sensor that value is meant for. Also, despite the fact that the type of signals has now increased to three, a single bit time slot is still sufficient to transfer any of the three signals when using a pulse based signaling approach as described in Section 6.6.

Cascading can thus be defined as:

Cascading: *If a sensor has values $v_1, v_2, ..., v_k$ to send to a neighboring sensor, it can send a single start signal, multiple intermediate signals corresponding to the values $v_1, v_2, ..., v_i$, and a stop signal corresponding to value v_{i+1}, where i is the minimum subscript such that*

$v_{i+2} < v_{i+1}$.



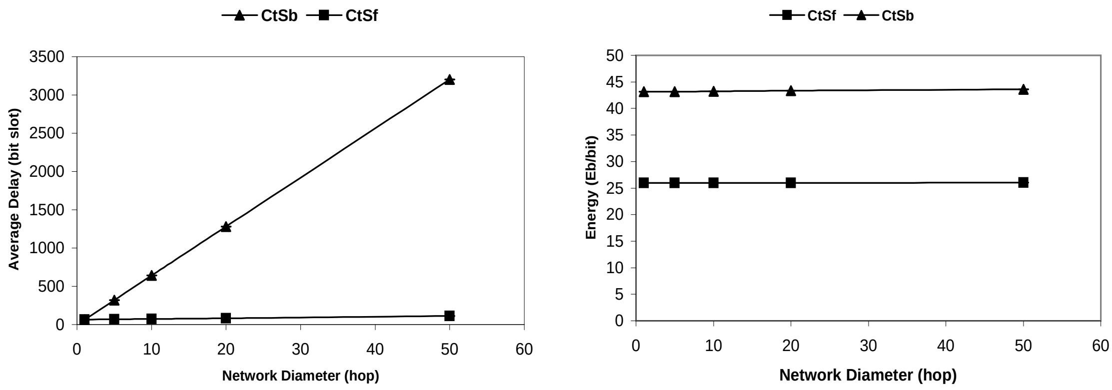
(a) Throughput Performance (b) Energy Performance

Figure 21: Impact of Cascading

Figures 21(a) and (b) present results for the throughput and energy consumption results for CtS_b and CtS_c respectively in a simple two node topology where a transmitter is sending information to a receiver. The values of the frames that the transmitter has to send to the receiver is generated randomly using uniform distribution. It can be seen that the throughput results for CtS_c increase by about 50% when compared to CtS_b. This increase is due to the cascading. Interestingly, the energy results are also better for CtS_c. This is because all but one value conveyed in a "cascade" is conveyed using effectively one signal. Hence, the average number of signals per value decreases to a value less than 2. Also notice that the throughput performance of both CtS_b and CtS_c degrade with data frame size, while their energy performance improve with data frame size. This is due to the longer CtS delay frames when the data frame size increases.

6.5.4 Fast-forwarding

Consider a sensor s_m sending a value 17 back to the sink along a path that includes other sensors s_{j1}, s_{j2}, and s_{j3} respectively. In both EbT and CtS_b, the value is conveyed first to s_{j1} completely, which then completely transfers the value to s_{j2}, and so on. Thus, considering CtS_b, and assuming that there are no other transmissions contending for the channel, the total time taken to transfer the value back to the sink is $4 * 17$, which is 68 bit time slots (when using EbT, it would have taken 20 bit time slots assuming 5-bit data frames).



(a) Throughput Performance (b) Energy Performance

Figure 22: Impact of Fast Forwarding

However, note that the sensor s_{j1} does not really need to wait for receiving the complete value when using CtS. Essentially, when s_{j1} receives the start signal at time slot t_i, it can go ahead and send its own start signal to sensor s_{j2} at time t_{i+1}. Similarly, when sensors s_{j2} and s_{j3} receive the start signals at time t_{i+1} and t_{i+2} respectively, they can go ahead and send their start signals in the next bit slot. Then, when sensor s_m eventually transmits the stop signal at t_{i+17}, sensor s_{j1} can immediately forward the stop signal at t_{i+18}. The sink will thus receive the stop signal at time slot t_{i+20}. Thus the same amount of information is now transferred in 20 bit time slots as opposed to 68 bit time slots.

More generally, for a h hop path from a sensor to the sink, and a value v that will take time $h * v$ to deliver back to the sink, such a forwarding technique will take time $v + h - 1$ to deliver the same information. We refer to such a forwarding technique as *fast-forwarding*, and CtS with fast forwarding as CtS_f. Note again that fast-forwarding is orthogonal in terms of logic to the earlier two techniques, and can be super-imposed on them relatively easily. In particular, if cascading is used, fast-forwarding will be applied to all three types of signals. Also, it is straightforward to observe that such physical overlapping of information transmission on subsequent contending hops is not feasible when EbT is used. Fast-forwarding can be formally defined as:

Fast-forwarding: *When a sensor sends information v back to the sink, sensors on the the intermediate hops can relay their respective start (and stop) signals on the slot next to the one they receive it on from a downstream sensor.*

Figures 22(a) and (b) show the performance results for CtS_f when compared to CtS_b in a scenario consisting of a linear chain of sensors where information is being sent from one end to the other (say the sink). Packets are transmitted hop-by-hop from the sender to the receiver. Using the CtS_b strategy, a relay node only transmits a packet to the next hop after receiving the entire packet (both start and stop signals). However, pipelined operations are still possible because a node can count for the next packet from the previous hop while transmitting the previous packet to the next hop. Using the CtS_f strategy, a node doesn't have to wait till it receives the entire packet to transmit the start signal, and hence, the end-to-end delay of delivering a data packet is much lower for CtS_f. It can be observed that the average end-to-end delay of transmitting one frame of CtS_b increases linearly with path length, which is caused by the "wait and forward" strategy CtS_b uses. The delay of CtS_f increases much slower with path length, since the end-to-end delay is dominated by the delay between start and stop signals, not the delay of relaying signals through multi-hops. The energy consumption of CtS_b is also worse since the total amount of time transmitting all the frames end-to-end is higher, hence each node spends more time being idle. Finally, It can be observed that the throughput of CtS_b falls down as the number of hops increases, as the delay is directly proportional to the number of hops traversed by the information. While the performance of CtS_f also decreases, it scales down far more gracefully due to the fast forwarding being performed. The average energy per node for transmitting each bit remains the same with different path lengths as the same number of signals are transmitted on every hop.

Fast-forwarding, while simple, can potentially be used to deliver better performance in other aspects of the communication. For example, the fast-forwarded signal can be used by the original downstream sender to act as an acknowledgement to its own transmission. Similarly, adaptation of the "waiting period" (as opposed to a constant waiting period of zero) - the period an upstream sensor waits before forwarding a signal based upon conditions such as channel loss-rate, volume of traffic handled by the sensor, etc., can help in further improving performance.

6.5.5 Other Optimization Possibilities

Until now we have discussed three optimization strategies: multiplexing, cascading and fast-forwarding. However, those strategies are by no means the only choices for CtS. We have considered other possible optimizations, and the feasibility and efficiency of those optimization strategies require further in-depth study.

For example, we have considered factorizing the number contained in a CtS frame into a list of prime factors and sending the factors to the receiver instead of the original number. The receiver only needs to calculate the product of all the factors to get the number to be sent. Using this scheme, given a specific sequence of values 3, 12, 21, and 30, the cascading can be performed using a "multiplicative" primitive by sending values 3, 4, 7, and 10, where the latter three values are used multiplicatively with 3 to re-create the original values. The total delay in this case turns out to be 10 bit slots, as opposed to the 30 time slots the original strategy would have incurred. In this way, the amount of time used to transmit each CtS frame is significantly reduced. However, problems such as how to find all the prime factors of a large number quickly, how to organize all the prime factors in a CtS frame efficiently, remain to be solved in order to practically realize the strategy.

In the above we only list an example of optimization strategy. Numerous improvements are possible due to the simplicity of the basic CtS strategy and the flexibility of its applications. We believe there are many potential possibilities yet to be explored.

6.6 Challenges

In the previous sections, we have introduced the CtS paradigm for Communication, while establishing that CtS can have considerable benefits both in throughput and energy consumption. However, the CtS mode of communication is significantly different from conventional energy based transmissions, and hence will require appropriate protocols tailored to the paradigm from the physical layer to several higher layers. In the rest of the section, we present the challenges that need to be addressed in perhaps the most important layer of the protocol stack - the medium access control (MAC) layer. We present the challenges in terms of distinct functionalities that need to be supported by the MAC layer. For each functionality, we describe the problem that needs to be solved, present a straw-man solution to the problem, and highlight the limitations of the solution and thereby expose research issues that need to be addressed. We also briefly outline what we believe will be the PHY layer requirements to facilitate the MAC protocol, and provide a short discussion on the impact of the CtS strategy on higher layers of the protocol stack. The goal of this section is thus to expose the research challenges that need to be investigated in order to realize CtS.

In the rest of the section, we refer to a MAC protocol supporting the CtS strategy as simply CtS-MAC. The specific challenges we discuss are framing, sequencing, addressing, error control, and contention resolution. Note that the solutions to the above problems will implicitly relax the assumptions (made in the previous sections) of framing, lossyness of the channel, synchronization of clocks, and addressability of signals. All of our discussions in the section are restricted to upstream communication, and we still assume higher layer protocol functionality such as routing to exist. Before we elaborate on the challenges associated with the development of CtS-MAC, we present the preliminaries of a typical sensor communication system consisting of the radio, physical layer communication paradigm, and clock that we believe will be required to realize CtS.

6.6.1 Radio Requirements Basics

Recall that a typical CtS frame consists of up to three signals - the start, stop, and intermediate signal. The radio available at the sender thus should be suitable for transmitting the above signals. In conventional communication systems that use EbT, a digital stream of bits is typically modulated to a higher frequency. To demodulate the bit stream correctly, the receiver has to first synchronize to the carrier frequency or phase of the data stream, and then interpret each frequency shift or phase shift as a *zero* or a *one*.

However, this synchronization overhead is usually a few tens of bits, if not longer [4]. Given the typical frame sizes usable with CtS - a few tens of bits(we discuss this issue in detail in the framing details of this section), such overhead clearly cannot be accommodated. Moreover, given the intentional delays that can occur between the start and stop signals, the receiver in fact might lose synchronization even between the two signals, and might have to be explicitly re-synchronized for both the signals. Hence the modulation scheme used for the underlying transmissions of start, stop, and intermediate signals has to take into account the above considerations.

We now proceed to elaborate on the specific design challenges associated with the realization of the CtS-MAC.

6.6.2 Challenge 1: Framing

The problem of framing has to do with determining the length of the messages the transmitter will send as a single unit to the receiver. This is the most basic design decision that needs to be addressed in any MAC layer protocol.

From the description of the CtS strategy, it is evident that the longer the silent interval between a pair of start-stop signals (or intermediate signals), the greater the energy savings, since on an average the same amount of energy is used to transmit more bits of information. However, the longer the silent interval, the greater the delay in conveying a specific value through the transmission. In general, the amount of information that can be conveyed in an interval of duration N bit slots is $log_2 N$. Hence, with increasing value for N, the throughput $\frac{log_2 N}{N}$ decreases. While the strategies presented in Section 6.5 do improve the overall network throughput performance, note that *for a single sensor attempting to transmit specific information, the length of the CtS frame will still determine the amount of delay needed for the transfer of that information.*

Hence, the decision on framing centers on how big the CtS frame size should be in terms of bit delay slots. In a traditional communication strategy using EbT, the frame size ranges from several hundred bytes to several thousand bytes. This is obviously not a feasible frame size for CtS: if the packet size is 100 bytes, the delay between start and stop signals should be 2^{800} bit slots! On the other hand, it is also not desirable to have a very small CtS frame size, since in this case, the energy savings brought by CtS scheme may not be significant enough, and may be even offset by the synchronization overhead required.

Our preliminary empirical evaluations have shown the practical CtS frame size to be $256 - 65536$ bit time slots, which translates to a raw data frame size of $8 - 16$ bits. The above frame size range, without any of the optimization strategies presented in Section 6.5, translates into 10Kbps and 100bps respectively for a 10Mbps raw data rate network. However, it remains to be determined how other environment factors will impact the choice of the frame size.

Note that in the frame size range obtained through the empirical evaluations, there is an obvious trade-off between the energy and throughput enjoyed by the sensors, depending upon the specific frame size chosen. However, there also exists another impact of the frame size, which is related to the contention resolution mechanism used. Briefly, when the frame sizes are large, the *silent periods* in CtS tend to be larger, and this in turn reduces the chances of collisions when simple contention resolution mechanisms are employed. We revisit this issue later in the section. Specifically, with respect to this impact, sensors might have to adapt the frame size depending upon the load conditions within their vicinity.

Another critical issue that needs to be addressed with respect to the framing strategy in CtS is the abstraction provided to the higher layers. Higher layer protocols ideally should not be burdened with the task of segmenting messages into the small sized frames required by CtS. One strategy is to still provide a "regular" link layer abstraction (with conventional frame sizes) to the higher layer, and hence higher layer protocols can provide CtS with "regular" sized frames. This "regular" frame can be split by CtS into frames of appropriate size, sent to the receiver back to back, and reassembled back into the corresponding "regular" link layer frame by the receiver, before being processed again.

6.6.3 Challenge 2: Addressing

While framing determines the length of the units of messages being transferred between a transmitter and its receiver, addressing is the critical functionality through which the transmitter indicates which neighbors a message is destined for, and hence which should act as a receiver.

Recall from section 6.5 the assumption about the addressability of the three CtS signals. Specifically, both the sender and the receiver need to be identifiable when a CtS frame is transmitted. While the conventional approach to embed the sender and receiver identifiers in the CtS frame is a possibility, such an approach would impose additional bit overheads on the already limited sized CtS frame.

A simplistic solution would be to record the original global source and destination addresses only in the "regular" link layer frame handled by CtS. However, for the CtS frames, locally computed addresses that distinguish only between sensors in a neighborhood will be encoded. A local coloring algorithm can be used to allocate the local addresses. In the coloring scheme, each sender-receiver pair forms a link, and each link is addressed by a color such that the sender and the receiver in a contention region can be uniquely identified by simply looking at the link color the packet carries. And the same color can be reused by sender-receiver pairs in different contention regions in the sensor network, and therefore, the length of such an address is determined by the minimum number of colors required to color the entire network (e.g. if 16 colors are needed, the length of sender-receiver address is 4 bits). In this fashion, we translate the addressing problem into a minimum coloring number problem.

Once the local addresses are determined, the signals can then be modulated with the address information. To modulate the signal with address information, each user is assigned a distinct code, and the code may be translated to a proper physical layer format.

While the above strategy is a high level overview of how the addressing issue can be handled, several open issues still remain to be addressed. How the coloring can be performed in a localized fashion, and how often it needs to be performed has to be addressed. More importantly, the specifics of how much of an overhead the addressing mechanism turns out to be needs evaluation, and the details of the mechanisms used need to be derived. Also, note that the above addressing strategy is only one of the possible ways of solving the addressing problems.

6.6.4 Challenge 3: Sequencing

Sequencing is the ability of a receiver to reconstruct information received from a transmitter in the same order that the transmitter had originally sent it, in the presence of reordering either due to losses or other reasons.

In a traditional communication scheme, a packet sequence number is required to cope with the problem of out of order packet delivery. The same situation may also occur in CtS. However, again due to the limited length of the CtS frame, adding sequence number to frames will result in significant reduction in throughput.

One possible solution is to not use a sequence number to any frame that is transmitted using CtS strategy. In such a set-up, the receiver might rely on the presence of sequence numbers in the "regular" link layer frame as in the traditional EbT scheme.

While this is a simple strategy and does away with the overhead issue, not having sequence numbers in the smaller CtS frames requires that all frames be delivered in order. Also, any completeness check can be performed only by waiting for all CtS frames to arrive at the point where the check is being performed. Most importantly, the strategy will require all CtS frames belonging to a "regular" link layer frame to be sent back to back by the transmitter, without any possibility of multiplexing CtS frames belonging to different regular link layer frames.

6.6.5 Challenge 4: Error Control

Error control is the problem of detecting errors in the received information, and possibly being able to recover from the errors.

Since CtS uses intervals between signals to infer information, the correctness of the scheme relies heavily on precise delivery of the start/stop/intermediate signals, and perfect synchronization of sender-receiver counting clocks during a single transmission [1]. There are thus two major error sources: external and internal, that affect the correctness of information delivery. In a wireless environment, white noise, multi-path fading and collisions may corrupt start/stop signals. In sender and receiver systems, in-deterministic processing, transmission and receiving delay cause unpredictable signal delivery time variations. Both can result in misinterpretation of the data inferred and we refer to these error sources as external interferences. On the other hand, within the sender and the receiver counting units, a certain degree of clock drift is unavoidable. This impacts the correctness of the

[1] Note that use of CtS does NOT require perennial synchronization between two sensors.

counting process itself and we call this error source as internal interference.

Both error sources adversely impact reliable data delivery of CtS, and due to its specific form of communication, well-established error control techniques traditionally used for normal EbT based communication schemes may not be applicable. Hence, the challenge is to properly incorporate error control measures into the system design, so as to achieve robust and efficient data delivery simultaneously.

A straw-man solution is for CtS-MAC to again rely solely on error control only at the granularity of the "regular" link layer frames, where traditional FEC (or ARQ) techniques such as Reed-Solomon code or maximum-erasure-burst-correcting code can be used to encode the raw data before transmission. The receiver, upon assembling back to back CtS frames from the same sender into a "regular" link layer frame performs error control at the level of that frame. However, the performance of such a simple strategy can be expected to be sub-optimal, and needs to be explored.

Another form of error control that can be explored is the use of *delay* as an error control mechanism. For example, consider an coding scheme where the value i is encoded as $2i - 1$ before being transmitted. Thus, if the transmitter has to transmit the value 5, the value 9 is actually transmitted. Note that under such a coding scheme, only *odd values* are valid under reception. Thus, if the value at the receiver is interpreted as 10 due to an error, the receiver immediately can detect the occurrence of an error. The trade-off in such a solution is the additional delay spent for the CtS communication. However, such a trade-off is akin to the trade-off of using more bits for coding in conventional EbT.

6.6.6 Challenge 5: Contention Resolution

Contention resolution is the problem of determining which of the sensors sharing a common channel in a neighborhood gets to transmit at any given point in time.

Recall that in CtS, the cascading strategy determines what a sensor locally wants to transmit, while the fast-forwarding strategy stipulates that a sensor forward immediately after receiving a start/stop or intermediate signal provided that certain conditions are met. Hence, the above two strategies do not influence the contention resolution process directly.

However, the multiplexing mechanism directly influences the contention resolution process, as when the sensors transmit information will directly impact potential collisions, and related performance issues.

Note that approaches such as *carrier-sensing* will have no application in the CtS strategy as the only signals transmitted are the start, stop, and intermediate signals, and such signals last merely for a bit slot duration.

While sophisticated contention resolution algorithms that depend upon the *values* each sensor wants to transmit can be devised (note that unlike in a traditional EbT based scheme where sensors have to look out for overlapping packet durations, in CtS sensors have to look out only for overlapping start-stop-intermediate signals, the positions of which are directly determined by the values), WSNs being resource constrained environments, such algorithms can turn out to be highly resource intensive.

On the other hand, a unique characteristic of CtS might allow for a relatively simple contention resolution scheme to be used very effectively. Recall that the empirically determined CtS data frame size is in the range of $8 - 16$ bits. In other words, CtS on an average conveys a data frame size of 12 bits using two signals. In other words, for k signals transmitted, CtS in effect transfers $6 * k$ bits of raw information. Now, consider the CtS (delay) frame size of $256 - 65536$ slots with an average size of 4096 slots (2^{12}). If the average number of signals transmitted during the 4096 slots is 800 signals, the total amount of information transferred can still be 2400 bits for an effective utilization of more than 50%. Now, note that the 800 signals translates to a average per bit slot access probability of only about 20%.

This characteristic of CtS can potentially be leveraged to require a relatively low access probability to generate reasonable throughput, and use a simple ALOHA medium access control strategy [50]. Since ALOHA's performance reasonably scales well at low loads of less than 20%, the performance of CtS does not suffer inordinately. At the same time, effective throughput utilization is maintained as justified above. Thus, a sensor, when it wants to transmit, merely goes ahead and transmits by picking a random bit slot within the frame slot window without regard to what other sensors in the vicinity are performing. If collisions occur, they will be detected by the error control strategy, which will then trigger retransmissions at the "regular" link layer frame level.

6.6.7 Challenge 6: Applications of Optimization Strategies

Optimization strategies improve the delay and throughput performance of the basic CtS strategy. However, they also complicate the CtS solution design. For instance, multiplexing requires a node to distinguish CtS frames coming from different flows in the case where the routes of multiple flows merge with each other. Under this circumstance, each CtS signal has to carry global address for upstream nodes to identify the specific flow this signal belongs to. On the other hand, if we do not use multiplexing, only local address is needed. We will discuss later how we avoid this overhead but still manage to leverage the benefit provided by multiplexing.

Another example of such complication is cascading. Cascading enhances the delay and throughput performance of the basic CtS strategy by introducing multiple intermediate signals between a pair of start and stop signals. Nevertheless, this strategy causes higher utilization of the wireless channel compared to the basic strategy and in turn results in higher probability of collisions. Moreover, the loss of one signal within a CtS frame entails the retransmission of all the signals (start, stop and intermediate) for this frame, indicating the higher impact of signal losses. Due to these reasons, an optimization strategy may not be applicable for all network configurations. To achieve the best performance, we have to carefully consider factors such as the availability of resources, the network load, the distribution of data sources etc. in order to come up with the best combination of CtS optimization strategies for a particular sensor network configuration.

6.6.8 Other Challenges

We have thus far discussed specific challenges pertaining to the medium access control layer when using CtS. While it is true that the most impact of using the CtS strategy will be on the MAC layer mechanisms, CtS will have an impact on other higher layer protocols too. In the rest of the section, we briefly discuss a few of these layers.

- **Energy Conserving Topology Control:** Topology control, in the context of sensor networks, refers to the problem of making a subset of the sensors in the network go to sleep in the interest of conserving energy [14]. The subset is chosen in a way such that the connectivity of the network is not compromised. Thus, while the sensors in the chosen subset go to sleep, the others stay awake maintaining connectivity. Such mechanisms can be used when using CtS strategy for communication, as the selection of the sensors itself is an orthogonal process. Note that any control packet exchanges required by such an algorithm will indeed be transmitted by the CtS mechanism.

- **Physical Layer:** One of the key differences between CtS and EbT is the amount of time a receiver will have to stay idle when waiting for a stop or intermediate signal after receiving a start signal. In conventional radios using EbT, while the energy required to sleep is several orders of magnitude lower than the energy required to transmit or receive, the energy when idle does not typically have similar discounts. Note that the consequent drawback (of additional energy consumption) is not different from a conventional EbT paradigm, where the sensors other than the chosen subset (see earlier discussion on topology control) have to be awake to ensure connectivity. However, the subtle difference is that the sensors in the EbT paradigm are staying awake to receive the next data packet (the arrival of which is not deterministic), while in the case of CtS, a sensor will stay awake to be able to receive any of the three signals, the arrival of which is not deterministic. The interesting question, however, is whether the simplicity of the communication strategy, where only one of three signals is sent or received, will enable the design and development of radios where considerable discounts in energy consumption can be achieved in idle state when compared to the transmit or the receive states.

- *Fusion:* Fusion is a primitive that is likely used in many wireless sensor networks to eliminate redundancy during the data gathering process. Essentially, when a sensor receives two related pieces of data X and Y from two downstream sensors, it fuses the data into Z, where the size of Z is less than the size of $X + Y$. While the fusion process can be performed in a network using CtS as-is at the "regular" link layer level, an interesting problem is whether the very mode of communication in CtS will allow for natural fusion of data to occur. For example, if both the start and stop signals arriving at a sensor from two different downstream sensors "collide" at exactly the same slots (meaning that the values are the same), can that be interpreted by the receiving sensor appropriately to perform fusion naturally?

6.7 Related Work

Two sets of solutions similar to CtS have been proposed in different application contexts. In this subsection we briefly introduce the two solutions and discuss their similarities and differences with CtS strategy.

6.7.1 Timing Channels

The possibility of using intervals between data transmissions to convey covert messages was first studied in [22]. In this approach, the duration of intervals between consecutive packets are translated into certain information in an alphabet at the sender. Receiver monitors the length of each duration, and translates those interval length back to the information delivered using the same alphabet. This scheme is similar to CtS in that both scheme use time interval to transmit information. However, there are several fundamental differences between them:

Timing channel approach is primarily proposed for secure communication: to use packet interval to infer covert messages, therefore, throughput and energy consumption of this scheme is not a major concern. Therefore, the achievable throughput using this timing channel is rather limited. While for CtS, the principal goal is to reduce energy consumption without significantly affecting overall throughput. The granularity of timing channel solution is on packet level, while for CtS, the granularity is on bit level. Because of this reason, it is difficult to achieve multiplexing for timing channel approach, since packet sizes are usually comparable to the interval lengths, or even bigger. While in CtS, the interval length between bits are usually much longer than bit length, therefore, multiplexing can be employed to improve the throughput performance significantly since the probability of collision is relatively low for a typical CtS application.

6.7.2 Modulation Schemes

In the context of modulation schemes, DPIM (digital pulse interval modulation) [19] has been proposed to convert bits into the number of time slots between two consecutive pulses. However, as a modulation scheme, the primary goal of this scheme is not to save energy. Since the interval between consecutive pulses is usually very small (around ten time slots), a sender generally cannot stop transmission during these intervals, therefore it's difficult to save energy. At the same time, it's difficult to apply similar optimization strategies as those of CtS to DPIM due to the short durations between pulses. Thus DPIM is still considered a conventional communication strategy: an energy based transmission scheme.

6.8 Conclusions

In this section we proposed a fundamentally different communication paradigm called *Communication through Silence* (CtS) to achieve energy-efficient communication without significant degradation on overall throughput in WSNs. The proposed scheme primarily uses time, along with a minimal amount of energy to deliver information between sensors. We analyzed in detail the primary energy-delay tradeoff inherent in this approach as well as other challenges related to realization of the proposed communication strategy. We presented several optimization strategies that can be used along with CtS, and can help in improving its throughput performance while retaining its energy benefits. Finally, we discussed challenges related to the design of a practical realization of CtS.

CHAPTER VII

PRACTICAL REALIZATION OF CTS: THE SOLUTION

In this section we present a solution suite that practically realizes the CtS strategy in wireless sensor networks. We first introduce the context of the solution, then discuss the specific details in terms of hardware, physical layer and MAC layer design, including addressing, framing, as well as synchronization solutions. Optimization strategies appropriate for the context are also included in the suite. To evaluate the performance of this solution, we compare it with the EbT scheme under the same network configurations using NS2. Simulation results are presented and discussed. Finally at the end of this section we discuss future research directions in CtS solution design.

7.1 Solution Context

For the proposed solution, we consider a typical multi-hop sensor network where a total of n sensors are randomly distributed in a sensor field. A subset or all of the sensors report data to a sink periodically using CtS strategy. The network is relatively static in that the topology does not change unless node failures occur. This configuration is representative of many sensor network applications such as environment monitoring and event tracking.

In a typical sensor network, due to the foremost concern of energy conservation, energy-saving measures are applied whenever possible. For the proposed solution, we assume a sensor network that implements any available energy-saving strategies (e.g. sleep scheduling, topology control) as long as they do not interfere with the CtS mechanisms. Of course, to ensure fairness, we compare the performance of CtS with that of EbT under the assumption that they both use the same energy-conserving strategies.

7.2 *Solution Details*

7.2.1 Overview

In this subsection we present every aspect of the CtS solution. We discuss in detail radio solution, counting, synchronization and framing solutions, addressing scheme, and synchronization scheme. For each solution, we first recap the problem, discuss its implications in CtS environment, and explain considerations in the design. After that we explain related technologies and their underlying principles that are relevant to the specific solution adopted by CtS. Finally we present the specific CtS mechanism, explaining what choice we have made for the mechanism and how the mechanism works. Other less complicated solutions, such as sequencing, optimization, and contention resolution, are introduced briefly at the end of this section.

7.2.2 PHY Solutions

7.2.2.1 Problem and Considerations

In a typical EbT sensor network setting, sensor nodes periodically go to sleep to conserve energy. This sleep state involves shutting down the wireless interface of a sensor node, and possibly its CPU and main oscillator too. During the sleep state, a sensor node is not able to receive any information from other sensors, since the wireless interface is not working. For a node to receive incoming packets, the wireless interface has to be in idle state, under which the power level is comparable (typically $1/10 - 1$ times) to that of transmitting and receiving states, as oppose to the sleep state, which consumes only $1/100 - 1/1000$ times the transmitting and receiving energy.

An important feature of the CtS strategy is the long duration a receiver spends waiting for signals from the sender. Due to the small amount of information delivered by a CtS frame, it is not practical for CtS nodes to consume energy in the schedule of CtS frame transmissions. Even the exchange of simple control signals becomes a luxury for CtS. As a result, the receiver cannot predict when a start or stop signal will arrive from the sender, hence cannot go to sleep if it has a conventional wireless interface. However, if the receiver stays awake all the time, a significant amount of energy will be wasted in idle states, which may even offset the energy savings of CtS.

To exploit the potential of CtS to the best, we leverage the recent advancement in the low-power radio technology domain [15], [23]-wakeup radio solutions to realize energy-efficient CtS operations. These techniques enable a CtS node to maintain a very low-power state while waiting for signals, and wake up for the reception only upon the arrival of an incoming signal. Wake-up radio interfaces have always been considered as an auxiliary interface to the main data radio interface, but in our solution, we use it as the primary communication interface. This leads to some interesting implications that we will discuss later.

7.2.2.2 Background and Context

Several radio solutions have been proposed in related work to achieve energy efficient operations in wireless sensor networks. [15], [48], [47] and [35] assume a second radio in sensor nodes to wake up its neighbors. This auxiliary radio uses much less power and is either always in a ready-to-receive state or wake up only on a low duty cycle. These wake-up radios consume very low power, typcially 1μW - 100μW, as oppose to 10mW for CDMA radios in monitoring mode. Nodes that wish to transmit send a wake-up signal to this node, which is captured by the wake-up radio, and in turn activates the primary data radio. The introduction of a second radio simplifies MAC design and achieves better energy efficient operations.

In its simplest form, a wake-up signal is just a busy tone; a binary signal without any information to identify the receiver. Every node receiving this busy tone becomes aware of an intended transmission, and turns on its data radio interface. This is not an efficient strategy. Augmenting schemes have been proposed to put non-receivers back to sleep state again(e.g. broadcasts a FILTER packet using the sender's data radio after the busy tone to explicitly identify the receiver [34]), but the energy consumed in state transitions and reception of the FILTER packets is still an undesirable overhead.

To solve this problem, advanced wake-up radios have be designed to embed short address information in the low-power wake-up signals, so that only the intended receiver is awakened, while all other nodes within the transmission range of the sender remain in sleep state [15], [23]. In the following we briefly explain two designs that embed addresses in the wake-up radio signal in different ways.

- *PicoNode wakeup radio*

  One of the goals in the PicoRadio project [6] is to design a low power standby transceiver specifically for the PicoNode [1]. This transceiver is capable of carrying addresses in radio signals using baseband coding. Coding in the baseband helps increase sensitivity, but at the expense of lower data rate. Since only addresses information need to be transmitted, and the radio needs to remain on anyways, data rate is not an important concern for this radio.

  PicoNodes in the same neighborhood are assigned unique bit sequences orthogonal to each other. Each transceiver contains a bank of correlators to match the addresses, and uses oversampling at baseband to achieve synchronization. Compared to the synchronization requirements of a data radio, the bit synchronization here is much easier since effectively only one bit (the wakeup sequence) need to be detected, and the sequence itself is a preamble.

- *Radio-triggered Wakeup Radio*

  The energy used to keep the wakeup radio always on, although small, is still an undesirable overhead. A power management scheme called radio-triggered wakeup [23] avoids this overhead by leveraging the energy of radio signals to activate a node with a passive radio interface. In this design, each node is equipped with a special hardware component - a radio-triggered circuit connected to one of the interrupts of the processor. Unlike Piconode radio circuits, this circuit itself has no power supply, thus does not consume any energy in wait state. When a sender wants to initiate a transmission, it emits a message with enough energy to trigger the passive receiver circuit, which in turn generates an interrupt to wake up the entire node. In addition,

[1] A low power wireless sensor node

the standby circuit should also be able to distinguish trigger signals intended for a particular receiver, in order not to wakeup on irrelevant signals. Both the requirements are fulfilled by the antenna design.

The length, size and shape of the radio antenna are specifically chosen to make it selective for radio signals: when an electromagnetic wave of a certain frequency arrives, the antenna is strongly activated, otherwise it's only weakly activated. The electrical energy collected by the antenna can be used to generate an output voltage to trigger the interrupt, waking up the sleeping node. From the experiment results in the paper, a wakeup signal transmitted at the power level of 10mW (typical for sensor nodes such as MICA2 mote) can generate an output signal with voltage higher than 0.6V at a receiver within 10 feet distance from the sender, enough to trigger an interrupt. To extend the effective range of the wakeup signal and reduce false alarm rate, active radio-triggered circuit with power supply is also designed. The additional energy consumption of the power supply can be made minimum (0.8% of the sleep mode current) if low-power amplifier is used.

Frequency selection based on antenna design is not flexible as it cannot adapt to the change of addresses assigned to each node. A scheme (radio-triggered ID) that exploits multiple frequencies to enhance the selectivity of radio-triggered hardware has been proposed. In this scheme, there are total of N frequencies usable by all the nodes. Each node is assigned a combination of several frequencies that is a subset of the N frequencies as its ID. Another node intending to wake up this node transmits a signal using this frequency combination [2]. Every node is equipped with N radio-triggered circuits, and each of the circuit can be activated by signals of a particular frequency from the N frequencies. For example, if 8 distinct frequencies are used in the network, and each node is assigned a combination of 4 frequencies, 105 unique frequency combinations can be used as local addresses. It is also possible to change

[2] Note that it is possible to transmit signals at multiple frequencies simultaneously using one transceiver only [25]

the ID of a node dynamically by programming logic gates connecting all the radio-triggered circuits. This scheme not only enhances the flexibility of ID assignment, but also increases the selectivity of radio-triggered wakeup.

7.2.2.3 CtS Mechanism

The PicoNode wakeup radio is a good candidate for CtS communication due to its two unique characteristics: the possibility for the radio to remain on all the time, and the capability of the radio signals to carry addresses. Since a CtS receiver cannot predict when the next signal will come, CtS radio has to be kept in idle state. This is unacceptable if a normal data radio is used, as the power consumption in idle state is comparable to that of transmitting and receiving states. While for wakeup radio this is not a problem at all. We also mentioned that in order to differentiate CtS signals, as well as the signal type (start, stop or ACK), each signal has to be able to carry addresses. Both requirements can be readily fulfilled by this radio.

Radio-triggered wakeup technique can also be readily applied to CtS communication, because of its addressing capability, and the nearly zero radio power consumption when nodes are not transmitting or receiving signals. Using this scheme, a particular signal type intended for a particular receiver can be coded into a combination of signal frequencies. From the outputs of the radio-triggered circuits, a CtS node can determine whether this signal is intended for itself, as well as the type of the signal. Compared to PicoNode wakeup radio scheme, this scheme enables even lower energy consumption, but at the cost of higher hardware complexity.

Unlike its EbT counterpart, in CtS node *wakeup radio is all that is necessary for communication, no companion data radio is needed.* This is because, in CtS, start and stop signals are the only information exchanged between sensor nodes, thus the role of the wakeup radio is promoted from auxiliary to primary. This implies very simple radio hardware as well as much longer battery life for CtS nodes.

Another interesting question to ask is: since both wakeup radios are capable of carrying data in the signal, why not use it for data communication directly? Two factors - low data rate and high error probability - determine that they are not good candidates for data communication. As a result of low system power consumption, the error rate of these radios is much higher (on the order of 10^{-1}) than that of a data radio. Therefore if wakeup radios are used to transmit a data packet, multiple retransmissions are usually required, and the number of retransmissions deteriorate quickly with data packet length. For the same amount of data, a normal data radio can transmit it with much higher data rate and lower error rate, and enter sleep state afterwards. Thus, for wakeup radios, the energy consumed per useful bit delivered may end up higher than that of data radio, and the throughput obviously will be much lower. That is why it is not a good idea to use wakeup radio for data communication. However, CtS mainly uses silent intervals rather than radio signals to transmit data. And the radio signals are only used to deliver very short address information. Considering the comparatively lower accuracy requirements of address information versus actual data, this error rate is acceptable. Therefore the drawbacks of wakeup radio in terms of data transmission is not a major concern in CtS, and is addressed by the reliability scheme we will discuss later.

In the above we discussed two radio solutions that can be used to implement CtS communications strategy. Those two schemes are not designed specifically for CtS, but their unique characteristics (low power consumption in idle states and addressing capability) make them ideal choices for CtS. With the continuous evolution of radio technologies, we believe there will be more advanced radio solutions coming up that are good candidates for CtS communications.

7.2.3 Counting, Synchronization and Framing Solutions

7.2.3.1 Problem and Considerations

Synchronization is essential for the accurate delivery of CtS frames: the counting clocks of the sender and the receiver need to be synchronized in order to count to the same number in a CtS frame's duration. Since we are considering multihop sensor networks, does this imply that the entire network has to be globally synchronized to ensure correctness? A naive solution may pose such a strict requirement, and in turn result in high message overhead and energy consumption. As global synchronization in sensor networks has been shown to be a very complicated and expensive process [20], in our solution, we manage to avoid global synchronization by leveraging optimization strategies so that synchronization is necessary only at the two ends of each flow (end-to-end synchronization). Furthermore, if we choose the CtS frame size appropriately and use counting clocks with enough precision, the CtS signals themselves can help maintain synchronization between the sender and the receiver. As a result no dedicated message exchange is necessary. We will explain the details of this solution later in this section

The size of a CtS frame determines its efficiency: the longer the frame, the more energy saving can be achieved, as we can deliver more data bits using the same two signals. Nevertheless, CtS frame size cannot be arbitrarily long due to two limiting factors: the application delay tolerance, and the counting clock drift. Within these two constraints, we want to be as greedy as possible to achieve maximum energy savings. As a result, the optimal CtS frame should be the longest possible as long as the delay requirement is not violated, and the synchronization between the sender and the receiver is maintained.

We assume the framing functionalities (dividing data packets into CtS frames and reassembling them back) to be implemented at the MAC layer. Network layer data packets carry global source and destination addresses while CtS frames only carry local addresses. Each data packet contains proper header and trailer information such that the packet can be recovered correctly at the destination from a series of CtS frames. It is recommended that appropriate coding schemes [36] [31] [64] be applied to the raw data packets depending on the channel conditions of packet delivery. These coding schemes provide extra protections for CtS frame contents given that the error detection and correction capability of CtS is limited by its energy constraints.

7.2.3.2 Background and Context

Counting is a primary focus of CtS solution design, as it determines how fast and accurate the CtS data can be delivered. Ideally, we want the counting frequency to be as fast as possible, because the faster the counting process, the lesser time is required to deliver the

Table 3: One-hop Basic CtS Throughput for Different Frame Sizes

Frame Size	10 bits	15 bits	20 bits
CtS Frames Delivered	20000 frames/sec	1250 frames/sec	20 frames/sec
Throughput	200Kbps	19Kbps	40bps

same amount of information, hence the higher the throughput. However, one factor limits the maximum counting frequency that can be used in a CtS node: energy consumption.

It is usually true that the energy consumption of an oscillator increases with counting frequency. Consequently sensor nodes typically do not use high frequency oscillators. But recent advancements in low-power electronic circuit design has enabled the application of high-frequency, low-power, and high precision oscillators in sensor networks. [43] presents a differential crystal oscillator that meet the tight stability specifications of a high performance radio system with a very low power consumption. Operated with a conventional AT 12.8MHz radio crystal, it consumes only 1μA current for a regulated 100mV output voltage. The stability of the oscillator is also very high: around $\pm 2ppm$ in room temperature.

7.2.3.3 CtS Mechanism

We can apply this oscillator circuit in CtS nodes, enabling high precision and low power counting functionality. Assuming a counting frequency of 10MHz, with a counting power consumption of only 0.1μW, following are the one-hop throughput results achievable by basic CtS for different frame sizes:

The table shows this counting frequency is enough to support throughput required for even heavily-loaded sensor networks, as long as the right frame size is chosen.

To ensure synchronization between the source and destination of a CtS frame, three factors need to be considered: the precision of the counting oscillator, the length of the multi-hop path, and the CtS frame size. The impact of CtS frame size is straightforward: The longer the frame size, the more vulnerable it is to counting errors. It is also relatively easy to quantify the impact of the precision of oscillators. For a given CtS frame of value n, as long as the relative drift (d_c) of the counting clocks between the source and the destination is less than $\frac{1}{n}$:

$$n * d_c < 1 \tag{41}$$

there will be no resulting counting errors. Accordingly, we have two degrees of freedom in synchronization design. First we should choose oscillators with the highest precision for a given sensor application energy and cost budget. If the resulted clock drift is still not low enough to eliminate counting errors, we can reduce the CtS frame size such that the above inequality is satisfied. For the oscillator circuit we discussed above, assuming a linear distortion of counting accuracy, no counting error result as long as the frame size is less than 22 bits (the maximum value to be counted to is 5×10^6).

Compared to the previous two factors, the impact of path length on synchronization is more profound. Signals from a source traverse multiple hops to reach the sink. On each hop, a signal could experience indeterministic delay resulting from transmission, propagation and processing.

Note that since the value of a CtS frame is determined by the duration between the start and stop signals, it is the delay jitter rather than the delay itself that affects counting synchronization. Several characteristics of the CtS solution help reduce the delay jitter of a CtS frame: 1) In fast-forwarding mode, intermediate nodes forward simple signals as soon as possible with very little processing, thus the indeterministic factor involved in processing is minimum. 2) The impact of transmission delay jitter is much smaller, because the size of a CtS signal is much smaller than the size of a EbT packet 3) Access delay which is caused by MAC layer contention and is a major delay component in most EbT communications, does not contribute to the total delay of CtS communications, since we use ALOHA scheme to deliver CtS signals. Currently we don't have a concrete physical implementation of the CtS solution to assess the impact of delay jitter. But for typical sensor nodes such as MICA2 Motes, the state-of-art hardware design enables delay jitters within 2μs between a pair of nodes [26]. Compared to normal sensor nodes, CtS nodes essentially eliminate most of the factors contributing to the jitter as listed above. Therefore we it's not difficult for CtS to achieve a delay jitter of less than 0.1μS per hop, enough to support a counting frequency of 10MHz.

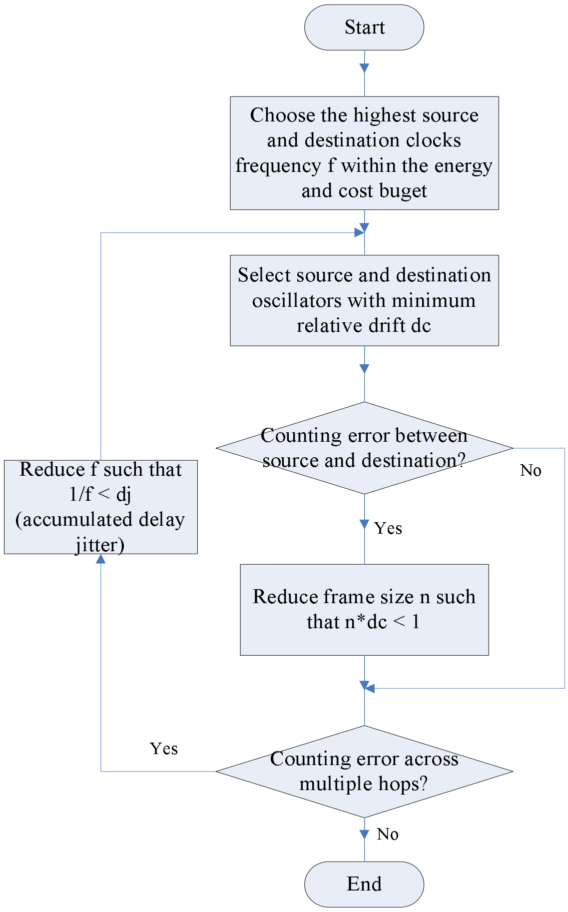


Figure 23: Synchronization Design Logic

Nevertheless, delay jitter generally accumulates along a CtS path, hence deteriorates with hop length. The counter measure is to choose shortest route possible for CtS frames in addition to minimizing the jitter at each hop. After both measures are adopted, if counting errors are still present, we can resort to reducing the counting frequency to finally eliminate the impact of delay jitter.

Figure 23 is a flow diagram illustrating the design logic of CtS synchronization solution. Here we assume fast-forwarding is used to deliver CtS signals, thus we only consider the clock drift between the source and the destination. Under this circumstance, the impact of multiple hops on the correctness of CtS frame, is simply the cumulative delay jitter along the path. We will discuss details related to the application of fast-forwarding later.

7.2.4 Addressing

7.2.4.1 Problem and Considerations

The easiest way for a receiver to recognize a packet from a specific sender is to embed the address of the sender in the data packet. Most routing solutions also require global addresses in the data packets to make routing decisions. This requirement places a considerable restriction on CtS frame size: the frame size has to be much longer than the global address length to amortize its overhead. Consider a sensor network of 1000 node, if we simply use the node ID of each sensor as the global address, the length of the address will be 10 bits ($2^{10} = 1024$). With such an address length, any CtS frame size less than 100 bits makes the address a significant overhead of the data frame, hence is not a feasible energy-efficient solution. But for a CtS frame size of 100 bits, it takes 2^{99} clock ticks on average to deliver a single frame, which is obviously a huge delay for any application no matter how fast the counting clock is.

The above example indicates that CtS frames should not carry global address. Given this limitation, a natural solution is to use local addresses instead. To ensure that the local address length is minimum, we use edge coloring algorithm to assign addresses to wireless links and make sure that the local addresses are unique within each sensor's neighborhood.

The use of local addresses also brings up the question about how to differentiate frames from different sources at a receiver. Because paths from different sources may converge before they reach the receiver, the local address of frames from different sources will be the same at the receiver. To avoid address conflicts, flows that share the same links cannot deliver frames simultaneously. We will show later how to use optimization strategies to improve throughput under this restriction.

7.2.4.2 Background and Context

To reduce local address overhead, the total number of local addresses used in the network should be minimum. In the sensor network settings we considered, local addresses are much shorter than global addresses because they can be reused in different contention regions. In our solution, we choose to assign addresses to links as opposed to nodes, because this assignment scheme results in less overhead. (Since a CtS frame has to carry the address of both the local sender and receiver when node-based addressing is used, the number of bits not actually used in the addresses tends to be more than a link-based addressing scheme).

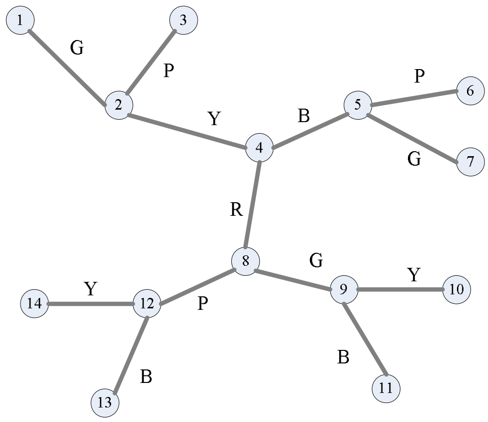


Figure 24: Minimum Link Color Assignment

If a link-based address scheme is used, the local address assigned to a link has to be unique within a two-hop neighborhood to avoid confusion. Figure 24 shows a link color assignment scheme to a network of 14 nodes and 13 links. To avoid address conflicts, links that are less than one hop apart cannot share the same color. For example, if we assign R(red) to the link between node 2 and 3 instead of P(purple), node 4 can overhear the transmission when node 2 transmits to 3 and will mistake this signal as a transmission from node 8 to itself. To avoid this conflict, the link between 8 and 4, which has a distance of less than two hops with any other links in the network, is assigned an unique color. While other links that are further apart can reuse the same color. For the assignment scheme shown in the figure, we use 5 colors in total: R(red), G(green), B(blue), Y(yellow) and P(purple) to distinguish 13 links, which can translate into an efficient addressing scheme. Therefore the primary concern in the addressing scheme is to minimize the required local address.

The problem of minimizing the total number of local link address in the network can be mapped into a minimum two-hop edge coloring problem in graph theory context. This problem can be formally stated as follows: *Given a graph G(V, E), where V is the set of vertices and E is the set of edges on the graph, find the set of colors C(G) with minimum size K(G) =* $|C(G)|$*, such that each edge on the graph can be assigned a color from C(G),* *and the color is unique within two hop distance.*

The minimum two-hop edge coloring problem has been studied in different contexts. One of them is the TDMA timeslot assignment problem in ad-hoc network settings [40] [42]. When TDMA MAC is used in an ad-hoc network, a local schedule need to be computed to assign slots to neighboring nodes so that each node can transmit data packets in its assigned timeslot to avoid conflicts. This timeslot has to be unique within the node's two-hop neighborhood, otherwise collisions will occur. In this context, minimum two-hop edge coloring generates TDMA schedules with shortest frame length (each frame contains exactly one timeslot for each source-destination pair in the neighborhood). From the above description, it's easy to see that our link address assignment problem is equivalent to the TDMA scheduling problem, which is NP-complete.

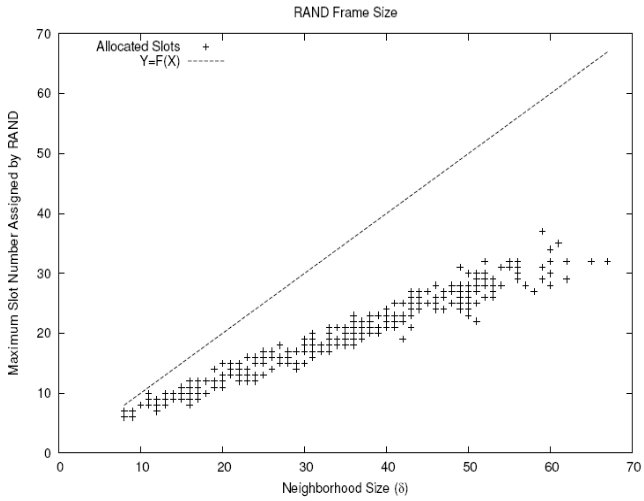


Figure 25: RAND Algorithm Performance

[40] proposes a well-known centralized TDMA scheduling algorithm (RAND) to approximate the optimal solution. RAND transforms TDMA scheduling into a node coloring problem in a graph, and sorts all the nodes in a random order. Colors are assigned to nodes using the random order without conflicts among neighboring nodes. The maximum slot number generated by RAND is $\delta + 1$, where δ is the maximum number of contending nodes in the network. RAND is a simple and efficient algorithm, but it cannot scale to large networks as it requires global knowledge to generate the assignment. Several decentralized

solutions [12, 42] have been proposed, among them [42] (DRAND) can produce any channel assignments generated by RAND but uses only local information with $O(\delta)$ time and message complexity.

In both RAND and DRAND, the number of time slots being assigned is bounded by $\delta + 1$. But in practice, this number is usually much smaller than the upper bound. Figure 25 extracted from [42] demonstrates the maximum slot number required as a function of neighborhood size. This graph shows that for a neighborhood of 30 nodes, around 15 time slots is enough to generate a conflict-free assignment. [40] also introduces other complicated centralized algorithms (MNF and PMNF) that give even better performance.

We can apply RAND or DRAND algorithm directly to CtS environment to generate link address assignment. The interesting question is which one should we use, RAND or DRAND? For TDMA scheduling problems, the answer is obviously DRAND, since it computes schedules distributively. In an ad hoc network, the transmission schedule for a certain neighborhood does not have to contain a timeslot for every node, since nodes without any data packet to transmit need not be included into the schedule. This helps reduce the schedule length and improve the overall throughput. But the set of nodes with packets to transmit usually changes quickly, as some nodes finish transmitting all their packets, and others start to accumulate packets. That's why the transmission schedule needs to be updated constantly. Centralized scheduling is not a practical solution here, because of the delay and the global-scale message exchange required. On the other hand, distributed scheduling algorithm can be invoked whenever necessary, on a local scale and with much less delay and message overhead. It is therefore a well-grounded choice for TDMA scheduling in ad hoc networks.

However, the same is not true for CtS. Compared to data flow generation, the local address assignment of nodes is relatively stable for a sensor network with low mobility. In CtS, the more obvious reason to update address assignment is node failures. In this case, a new local address assignment may result in lesser address bits being used. However, in order to reduce 1 address bit, at least half of the nodes within the neighborhood should fail, which does not occur frequently and may even result in network partition. Distributed algorithms also require control message exchanges between local nodes to update their state information and reach a consensus in the assignment [42]. Considering all the above disadvantages of distributed algorithms, centralized assignment algorithms are better choices for CtS.

7.2.4.3 CtS Mechanism

The application of RAND or other centralized scheme to CtS is rather straightforward. The address assignment algorithm is invoked only at the network startup. During this phase the sink collects information about the network topology, and compute the minimum link address assignment in a centralized fashion. After that the sink propagate the assignment to the entire network, which stays unchanged most of the time. Recomputation of the address assignments may be used when the network topology changes significantly, but this process is infrequent and invoked only when necessary. The following pseudo code describes the address assignment process.

Input: 1) Sensor network topology, 2) Transmission range

Output: A link address assignment scheme

Begin:

1. Generate graph G(V,E) from the given topology and transmission range

2. **For** every edge e in E **do**

3.   Color(e) $\leftarrow$ 0

4. **For** labels l from 1 through $|V|$ **do**

5.   Pick unlabelled vertex u at random

6.   Label(u)$\leftarrow$ l

7. **For** j = l_{max} to l_{min} **do**

8.   Let u $\leftarrow$ vertex with label j

9.   **For** each edge e incident on u **do**

10.   Color(e) $\leftarrow$ the first color not assigned to any of its neighboring links within two hops

11. Find the total number of colors Nc

12. Compute the address length L = $\log_2 Nc$

13. Map the color of each link to an unique address of length L

end

Also note that we need to distinguish two types of signals on each link: start and stop. Thus the total number of addresses required should be doubled. For a neighborhood of 30 nodes, around 30 addresses are needed. This translates to 4 bits of address information that need to be carried in each CtS signal. This requirement is easy to fulfill using either of the radio technologies we introduced in 7.2.2.

7.2.5 Reliability

7.2.5.1 Problem and Considerations

Three factors contribute to packet errors in a CtS environment: signal loss, timing inaccuracy and misinterpretation of CtS signals. We briefly discuss here how we address each of the error sources. Timing inaccuracy occurs when the sender and the receiver of a CtS frame are not perfectly synchronized. However, this inaccuracy does not necessarily lead to CtS frame errors: as long as the clock difference between the sender and the receiver is less than one for the duration of the CtS frame, they still count to the same number, hence the correctness of the CtS data is not compromised. Thus in order to eliminate CtS errors caused by timing inaccuracy, we should focus on keeping the clock discrepancy between the sender and the receiver within an acceptable range, rather than achieving perfect synchronization.

A CtS signal transmitted between the sender and the receiver may be lost due to lossy links or collisions. We use a simple ACK scheme to detect and recover lost signals. An end-to-end reliability is implemented at the MAC layer such that each CtS frame sent by the source is acknowledge by the destination directly. For each pair of start and stop signals received correctly, the receiver sends back an ACK to the sender. Coarse-grained timer is used in conjunction with ACK signals to avoid misinterpretation of signals. We will get into the details of the ACK scheme later.

CtS signals transmitted within a neighborhood are differentiated by embedding the address of the local sender and receiver in the signal. As a result, compared to EbT, where distortion and interference of radio signals mainly cause data errors in a packet, in CtS the most possible type of errors observed is bad signal addresses. A CtS signal with wrong address may be received by the receiver that is not intended for, or discarded by all nodes in its neighborhood. In either cases from the destination's perspective, the signal is considered as lost. Hence the entire frame associated with this signal can be recovered by the ACK scheme we discussed earlier. From other nodes' perspective, misinterpreted signals may disrupt the communication of other flows in the neighborhood. In this case, a combination of ACK scheme and error correction scheme at higher layer can correct this problem.

At the network level, reliability scheme should also address node failure problems that occur very often in sensor networks. A node failure affects the correctness of CtS data only when the node is on the routing path of an active CtS flow. In this case, rerouting scheme can be used to find an alternate path between the source and destination, and error correction measures at higher layer are used to recover CtS packets.

7.2.5.2 Background and Context

Two design choices have to be made for the CtS ACK scheme: whether to use end-to-end reliability or hop-by-hop reliability, and the layer at which the reliability scheme is implemented. In EbT communications, hop-by-hop reliability is implemented at the link layer, and end-to-end reliability is implemented at the transport layer. However it is difficult for CtS to adopt this model because of its unique characteristics.

In the case of hop-by-hop reliability, each CtS node on a path ACKs a CtS frame after receiving the start and stop signal pair. If any of the start/stop/ACK signal is lost, timeouts at both the local sender and the receiver will be incurred. And the CtS frame will be retransmitted by the local sender. Since we use fast-forwarding in this solution, a start signal usually arrives at the destination (the sink) long before the source transmits the corresponding stop signal. If a timeout occurs at an intermediate hop, all the upstream links will also experience timeouts(despite the fact that all start/stop/ACK signals of this CtS frame may have been exchanged correctly, when the timeout is caused by ACK loss in a intermediate hop). As a result retransmitted signals have to propagate all the way to the sink from the lossy link.

Note that downstream nodes still continue to transmit new signals in the mean time because they are not aware of the losses occurring at upstream links. Those new frames are held at the node where the first time out occurs until the previous frame has been retransmitted and ACKed properly. Consequently, hop-by-hop reliability scheme entails intermediate nodes to be able to translate CtS frames into data values, hold those values in its buffer, translate the values back into CtS frames and send them when necessary. As the losses on the paths accumulate, intermediate nodes have to store more and more CtS frames. At a certain point, flow control has to kick in to avoid buffer overflows at those nodes. This results in a reliability scheme with high complexity, hence is not a desirable choice.

If an end-to-end ACK scheme is used instead, ACKs are exchanged directly between the destination and the source, while intermediate nodes only forward start/stop/ACK signals blindly. They do not have to understand the reliability semantics, hence the logic implemented in each node is much simpler. Moreover, intermediate nodes do not have to translate or store CtS frames, thus don't need to maintain any fine-grained timer or buffer space for data. This greatly reduces the CtS node's operation complexity, especially when a node is traversed by multiple flows.

A probable disadvantage of the end-to-end scheme is that if a signal is lost in an intermediate hop, it has to be retransmitted through all the links on the path. But this is not a significant drawback, since the CtS scheme usually works at low network load condition, therefore signal loss is not frequent. Furthermore, as we described earlier, hop-by-hop reliability cannot completely avoid redundant signal retransmissions either.

Another possible disadvantage for end-to-end reliability is that counting errors will accumulate along the path, while for hop-by-hop reliability this is not true. In hop-by-hop reliability, CtS frames are essentially converted into data at each node and reconverted back to frames and transmitted to the next hop, therefore counting errors won't accumulate. Nonetheless, end-to-end reliability itself helps reduce the counting error by reducing processing time at each intermediate hops, since each intermediate node only needs to forward signals blindly. Compared to the duration of the CtS frame, the time used to forward CtS signals through multiple hops is usually much smaller. As a result the delay of delivering a CtS frame through multiple hops using fast-forwarding is similar to the delay of delivering it across a single hop. Furthermore, we expect that under most applications, the accumulated delay jitter along multihop paths are still low enough to not cause any misinterpretation of CtS frames. For some restrictive CtS applications, we can always fall back to hop-by-hop reliability model to eliminate counting errors.

Because of the use of end-to-end reliability scheme, and the absence of out-of-order frame delivery in CtS communication (we will explain the reasons later), the necessity of a transport layer is eliminated. As a byproduct, this design results in a simpler protocol stack for CtS, which is especially beneficial for energy-constrained sensor nodes.

7.2.5.3 CtS Mechanism

We design an end-to-end acknowledgement scheme to ensure the reliable delivery of CtS signals over multiple hops. With this scheme, for each start/stop signal pair received correctly, the sink sends an ACK signal along the same reverse path to the source. Note that the addition of the ACK signal does not increase the need for local addresses, because the ACK signal is always transmitted in the reverse direction (from the receiver to sender), therefore can reuse the address of the start or stop signals. A timer is also needed for the receiver to identify different types of signal losses and retransmitted signals.

Now we explain how signal losses are handled by the ACK scheme.

Any of the start, stop or ACK signal loss triggers timeout at the sender. Since the sender needs to retransmit both the start and stop signals, there is no need for a sender to differentiate these losses. The sender sets its timer to a value Δ after transmitting a start signal, and resets its timer upon the arrival of an ACK. Δ is the maximum value of a CtS frame for a given frame size, thus the delay between a start and stop signal pair should be less than this value. If no ACK arrives before the timer expires, the sender infers a loss of any of the three signals and retransmit the previous frame. Otherwise the next frame is transmitted.

We require that a start signal for a new frame has to be transmitted after δ_1 time interval and before δ_2 time interval after receiving ACK of the previous frame. A retransmitted start signal should be sent out immediately after the timer with value Δ expires. The δ_1 is used as a guarded time interval to avoid confusion when the length of a CtS frame is close to the maximum length. We will see how the mechanism works in the sequence diagram below.

The receiver needs to distinguish start, stop and ACK losses so as to interpret the next incoming signal correctly.

- *Stop loss:* Stop loss is easy to recognize. Since the receiver cannot predict when a stop signal will arrive, it sets the timer to Δ after receiving a start signal from the sender. If no stop signal arrives before the timer expires, the receiver can infer the stop loss, consequently expect the retransmitted start signal within δ_1 interval after

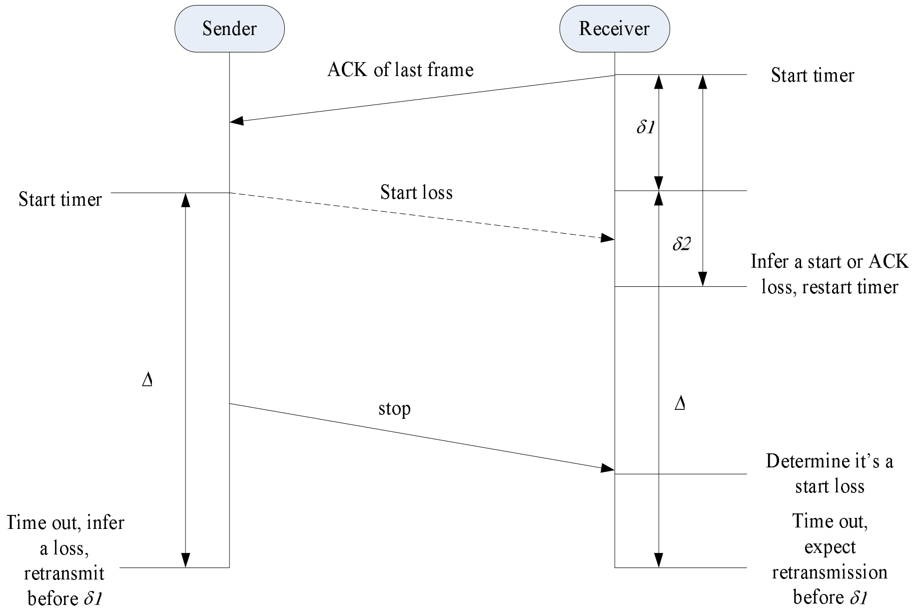


Figure 26: Time Sequence Diagram (a): Start Loss

the time out.

- *Start loss and ACK loss:* The identification of start and ACK loss is more complicated. The receiver starts its timer with value δ_1 after sending out an ACK. If no start signal comes between δ_1 and δ_2, there are two possibilities: either the next start signal is lost, or the previous ACK is lost and the sender experiences a timeout. The two scenarios can be differentiated by the next signal the receiver gets: if the next signal is a stop, the former possibility is true, otherwise the later is true.

  A special case is when the CtS frame length L is close to the maximum frame size Δ, in which case the sender times out immediately after the time the ACK should arrive. Under this circumstance, the sender's timing for start signals can help the receiver identify new and repeated transmissions. If the receiver receives a start signal before δ_1 interval after the ACK, it can infer an ACK loss, and consider this start as a retransmission. On the other hand, if the start arrives between δ_1 and δ_2, the start signal is interpreted as a new transmission. Without this mechanism, the receiver cannot differentiate theses two cases.

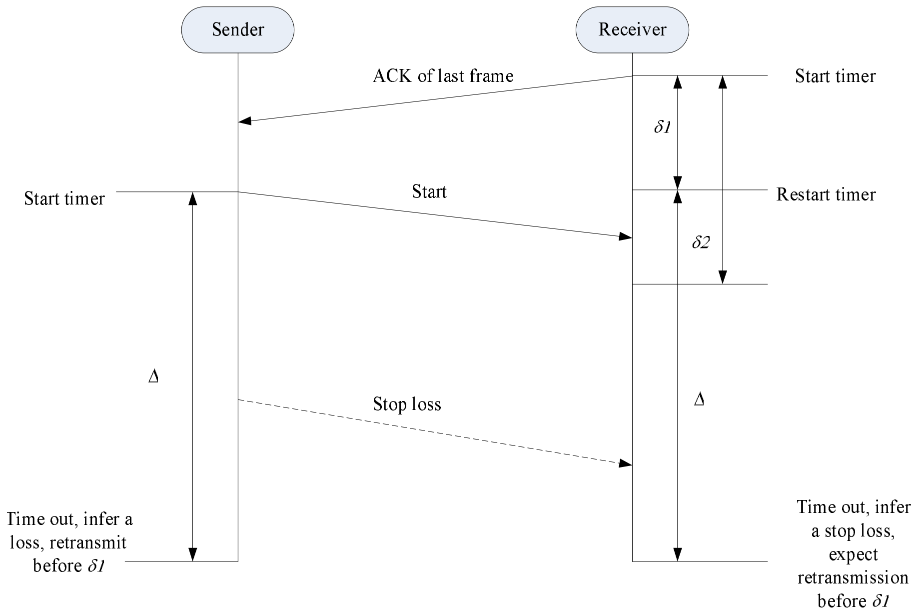


Figure 27: Time Sequence Diagram (b): Stop Loss

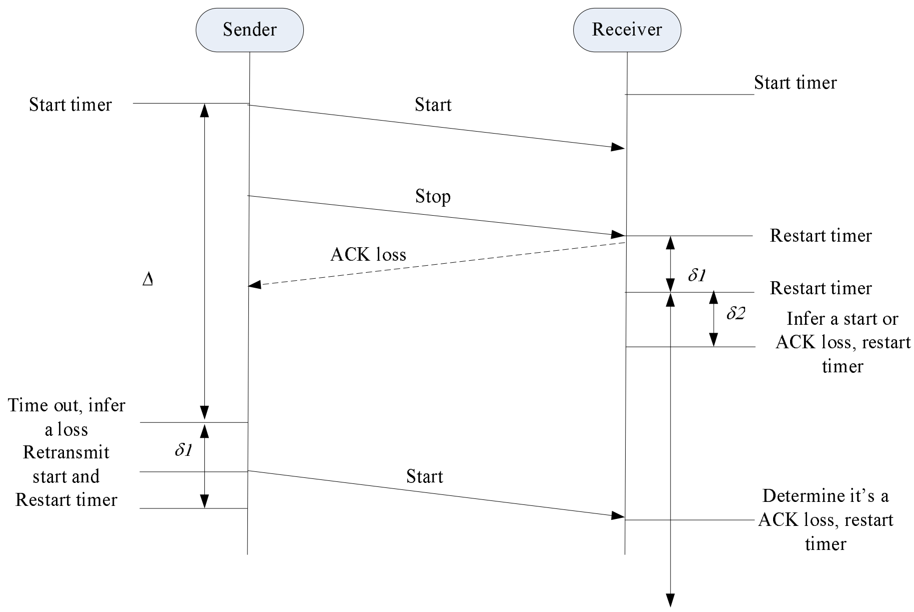


Figure 28: Time Sequence Diagram (c): ACK Loss

We show the time sequence diagrams of the three types of losses in figures 26, 27 and 28 [3]. Note in the above discussions and illustrations we do not consider the delay of transmitting signals along multihop paths. This simplification does not affect the correctness of the sequence. For real applications, we can simply adjust the values of timer according to the estimated delay between the sender and the receiver.

In the rare case of consecutive signal losses [4] , the logic of the receiver may be confused, therefore will consider some of the signals received as unexpected. Under these circumstances, the receiver can infer a mismatch of signal interpretations at the sender and the receiver, and send a restart signal (another signal type in the reverse direction). This signal enables the sender and the receiver to resynchronize their timers, and the sender starts retransmission from the first CtS frame it has not received ACK for. Coding scheme at the packet level can also be leveraged to detect and correct errors that are not handled by the ACK scheme.

7.2.6 Other Solutions

7.2.6.1 Sequencing

In conventional EbT communications, sequence number is usually included in each data packet to facilitate the reassembly of higher level data in the presence of out of order delivery. Out of order delivery is mainly caused by two reasons: multiple routing paths and retransmissions. In the former case, data packets from the same source may be delivered via different paths. This in turn results in packets leaving the sender later arriving at the receiver earlier due to the different delays on different paths. In the later case, when a data packet is lost during its initial delivery, reliable transport protocols such as TCP can detect the loss and retransmit this packet. Yet during the loss detection phase, a lot of new packets are transmitted in order to make full utilization of the network pipe bandwidth. As a result, the retransmitted data packet becomes out of order packet. In both cases, the sequence number carried by a data packet can be used to recover the order of the packets

[3] Figures are not drawn to scale

[4] This is rare because the channel utilization of CtS is usually very low, and the time intervals between a pair of start and stop signals are usually very long

correctly.

Similar to global addresses, sequence numbers also incur non-negligible overhead in CtS frames, especially when the duration of a data flow is long. Fortunately, sequence number in a data packet is dispensable as long as the causes of out-of-order delivery are eliminated. To avoid multiple routing paths, we choose a connection-oriented routing solution for CtS, in which the routing path for a particular CtS flow during its lifetime remains the same. This solution is reasonable since we mainly consider sensor networks that are relatively static. However, sensor node failures may still cause routing changes. Under these circumstances, rerouting mechanisms should be invoked to find a new route. This problem is part of our future investigation.

To eliminate out-of-order packets caused by retransmissions, we require that a new packet can be sent from a sender only after all the previous packets have been received correctly by the receiver. This is consistent with the end-to-end reliability scheme we discussed earlier. From the first glance, this approach seems inefficient: at any given time, there can be only one packet/acknowlegement in transit on a network path, which seems to be a waste of the available bandwidth resources. This is true for EbT-style communication, but for CtS with fast-forwarding, the delay of transmitting a CtS frame end-to-end is close to one-hop delay. Therefore this packet-by-packet delivery scheme does not cause much degradation in network resource utilization compared to a pipelined delivery scheme.

7.2.6.2 Contention Resolution

We use a simple ALOHA scheme for the contention resolution of CtS in which a node simply transmits signals when necessary, without any carrier sensing or control signal exchange. This scheme minimizes the overhead involved in contention resolution, and is optimal when the network load is relatively low [50]. In a CtS environment the channel utilization is inherently low due to the predominant silence interval for each CtS frame, and it is not feasible to apply control signals to achieve contention resolution. Therefore ALOHA scheme is an ideal choice for CtS.

7.2.6.3 Optimizations

We have discussed several optimization strategies in section 6.5, including multiplexing, fast-forwarding, and cascading. The applicability of these optimization strategies depends on the application context, and resource availability. In the sensor network environment we considered, fast-forwarding is essential as it can significantly reduce the end-to-end delay of CtS frames over multiple hops.

Multiplexing helps improve CtS throughput, especially when multiple flows intersect at a particular node. In the case of EbT, such a node becomes the bottleneck, as a result all flows suffer much lower throughput. This problem does not occur to CtS, because each CtS frame is characterized by a long silent period, during which signals of other CtS frames can be delivered by the same node. However, notice that if the outgoing links of two flows are the same, the upstream node cannot differentiate frames from the two flows as only local addresses are used. Since global addresses impose a considerable overhead on size-limited CtS frame, we use multiplexing only when the incoming and outgoing links of intersecting flows are separate. In many-to-one communication as we consider here, multiplexing does not help much because all flows eventually converge at the sink. But for many-to-many communication, multiplexing can improve the overall throughput greatly.

If multiplexing is not used, every node on the path of a particular flow can only forward CtS frames this flow, the long silent duration between start and stop signals are not efficiently utilized. Another optimization strategy - cascading - can be used to alleviate this problem. By compacting consecutive CtS frames that are monotonically increasing in values, the throughput of a CtS flow can be increased up to 50% as indicated in section 6.5. Nonetheless, the application of cascading requires the use of an extra signal type: intermediate signal. At the same time, the impact of signal losses is more severe (entails the retransmission of all start/intermediate/stop signals) when cascading is used.

7.3 Performance Evaluation

To evaluate the performance of the CtS solution, we implement it in NS2, and use simulations to compare the energy and throughput of CtS with those of EbT scheme. In the simulations we consider two network configurations: a single-hop network in which all sensors transmit to the sink directly and a multihop network in which sensors transmit data to the sink via multihop paths.

In the single-hop network, multiplexing is used to improve the overall throughput of CtS strategy so that the sink can receive data from multiple sensor nodes at the same time. In the multi-hop network, fast-forwarding is used to improve the end-to-end throughput of each CtS flow. If multiplexing of CtS signals are used in intermediate nodes, flows tend to converge at nodes that are close to sink. As a result global addresses have to be used to identify CtS signals from different downstream nodes. To make the CtS signal as simple as possible, we use multiplexing only at the last hop (in the sink's neighborhood).

We assume CtS nodes use one of the wakeup radios we discussed earlier, and EbT nodes use a data radio with 802.11 MAC. The typical energy level of wakeup radios is $10 - 100\mu$w [15]. While for the data radio, we assume a low power radio that are commonly used in sensor nodes such as MICA2 [2]. This data radio uses Chipcon cc1000 low-power transceiver [1], which has a transmission power level of 10mW. Although this power level is much lower than that of a typical PC wireless interface card (the transmission power of which is of the order of 1000mW), it is still significantly higher than the power level of wakeup radios.

To ensure fair comparison, and highlight the energy differences resulting from the underlying communication strategy, we assume each EbT node is also equipped with a wakeup radio. With the help of the wakeup radio, the node can remain in sleep state whenever it's not transmitting or receiving, thus no wasting of energy in idle states. Therefore in the simulations we only count the energy used in transmission and reception, and do not include any idle energy in the calculation.

Since the energy level used by the wakeup radio is several order's lower than that of a data radio, we ignore the energy consumption of wakeup radios for EbT nodes. According to the energy level difference between a data radio and a wakeup radio, the power consumed transmitting one bit using data radio should be $100 - 1000$ times the power consumed transmitting one bit using wakeup radio. However without a real implementation we cannot have an accurate estimation about the actual energy consumption of a CtS radio. Also, considering the overheads that may be involved in transmitting CtS signals (e.g. the energy consumed in state transitions), in the simulations we simply assume that each CtS signal costs the same amount of energy as a EbT bit. The above two assumptions makes the energy results of CtS worse than its actual performance. But we show later in the results that the energy performance of CtS is significantly better even with these discount factors.

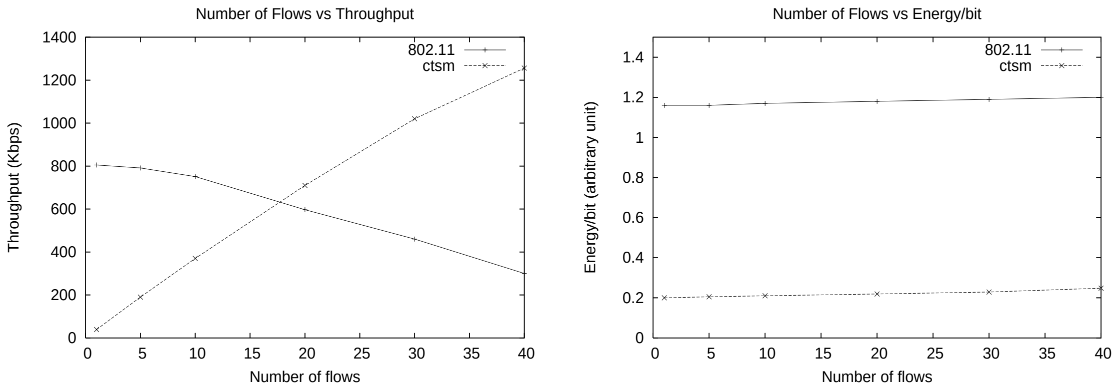
We assume the data rate of EbT radio is 2Mbps, which equals to the counting frequency of CtS nodes. In terms of counting energy, given the available low energy oscillator circuit we introduced earlier, and the necessity of keeping an oscillator always on even in an EbT node, we ignore the contribution of counting energy to the total energy consumption.

To focus on the effectiveness of MAC layer protocols and CtS mechanisms in this performance evaluation, we assume simple static routing. At the beginning of each simulation run, routes from each source are computed, and remain unchanged during the simulation. How to come up with routes for CtS and how to change routes dynamically is a subject of future research.

7.3.1 CtS Performance in A Single Hop Network

In this set of simulations, we evaluate the performance of CtS with multiplexing in a single hop environment. In each simulation, 150 nodes are distributed randomly in a 200x200m grid. The transmission range of sensor nodes is big enough such that each sensor can talk to the sink directly. For EbT scenarios, 802.11 MAC is used to coordinate the data packet transmissions, while for CtS, CtS MAC with ALOHA access scheme is used. With multiplexing strategy, multiple senders can transmit to the sink at the same time, leveraging the silent intervals between start and stop signals.

The data packet size of EbT is 1000 bytes and the CtS frame size is 10 bits per frame, which remains constant for all the simulations. The total number of simultaneous flows in the network is increased from 1 to 40 across simulations. For EbT scenarios, the rate of individual flow is set to a value such that the network is always saturated. This data rate configuration is specifically designed to explore the capacity of the network. Each data point on the graphs shown below are generated based on results from 10 random seeds, and is displayed with 95% confidence interval.



(a) Throughput Performance (b) Energy Performance

Figure 29: Performance Comparison between CtS with Multiplexing and EbT

Figure 29(a) shows the aggregate throughput of CtS and EbT as a function of total number of flows. From the figure we observe that the aggregate throughput of EbT is highest when there is only one flow in the network, and decreases as flow number increases. This decrease is caused by the distributed inefficiency of the 802.11 MAC, and is an artifact of the many-to-one communication typical in sensor networks. In a single hop network, each sensor can hear the transmissions of other sensors, therefore can invoke collision avoidance mechanism if necessary. When the number of flows in the network increases, sensor nodes spend most of the time in backoff, causing less efficient channel utilization. As a result the aggregate throughput decreases monotonically despite the increasing number of flows.

On the contrary, the CtS throughput increases monotonically with flow numbers because of the increased channel utilization enabled by multiplexing. When there are 20 flows in the network, the aggregate CtS throughput is already close to 700kbps, higher than the aggregate throughput of EbT. Moreover, as the total flow number is increased to 40, the throughput of CtS is as high as 1.2Mbps, which is 4 times higher than the EbT throughput at the same flow number.

The results indicate that CtS with multiplexing can greatly increase the aggregate throughput at the sink. This indicates CtS as a cure for the sink bandwidth bottleneck problem that causes low throughput in many sensor network applications. This problem usually occurs in sensor networks with relatively high data rate sensors such as image sensors. In these applications the task of the sink is to collect data from all the sensor as fast as possible before processing and integrating them. But when there are many sensor nodes in the network, throughput at the sink decreases due to increasing collisions, causing undesirable delay at the application layer. However, if we apply CtS with multiplexing in this scenario, a much higher aggregate throughput can be achieved and the application delay requirements can be satisfied. This observation implies that CtS is not only a solution to reduce energy consumption, but can also be a measure to improve throughput for specific network conditions.

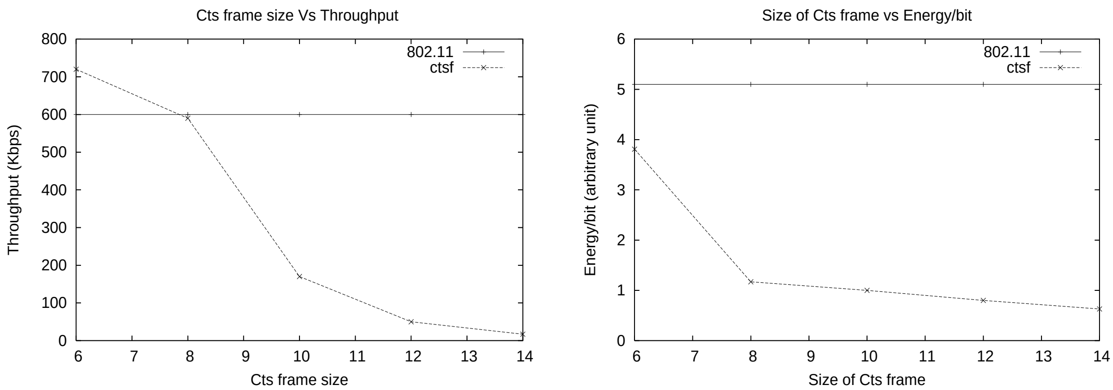
Figure 29(b) shows the energy consumed per data bit delivered to sink for CtS and EbT as a function of total number of flows. Both curves are almost flat, indicating that there are very few collisions despite the increased flow number. This happens in 802.11 because nodes end up spending most time in backoff when more nodes attempt to transmit within the same contention region. But for CtS, the number of collisions is naturally low, because nodes spend most of the time in silence even when there are delivering data. As to the energy levels of the two scheme, for each data bit transmitted, CtS uses only 1/6 the amount of energy that EbT uses. This result highlights the advantages of CtS as an energy-saving communication strategy.

7.3.2 CtS Performance in A MultiHop Network

In this set of simulations, we evaluate the performance of CtS with fast-forwarding and multiplexing at the last hop in a multihop environment. In each simulation, 150 nodes are uniformly distributed in a 1000x1000m grid, and data sources are selected randomly from the sensor nodes. The average path length from sources to sink is 4 to 5 hops . For EbT scenario, 802.11 MAC is used to coordinate the data packet transmission, while for CtS, CtS MAC with ALOHA access scheme is used.

To improve the end-to-end throughput of CtS, we use fastforwarding strategy to deliver CtS signals across multiple hops. Reliability scheme discussed earlier is applied to ensure the correctness of CtS frames. Since we assume CtS signals only carry local addresses, multiplexing is difficult to achieve at intermediate nodes. As a result, we only use multiplexing at the last hop (i.e. in the sink's neighborhood).

Given this setup, the number of simultaneous flows that can coexist in the network is the same as the sink's neighborhood size. Therefore, in the simulations, if multiple flows pass through the same node, only one is kept and all the others are suppressed. This is accomplished during the route computation process. In real scenarios flow control signals can be delivered downstream to temporarily stop those flows. Those flows can either choose other routes, or reattempt the transmission upon receiving a resume signal from the intermediate node after the current flow stops. We will discuss the rerouting and flow control scheme in 7.4 section. Note that the same problem exists for EbT too as a result of many-to-one communication.



(a) Throughput Performance (b) Energy Performance

Figure 30: Performance Comparison between CtS with Multiplexing and EbT

Since the number of flows is fixed for a given topology, in this set of simulations we study the energy and throughput performance of CtS as a function of frame size. And the total number of flows in each simulation is chosen to be the same as the sink's neighborhood size. Figure 30(a) shows the aggregate throughput of CtS when the frame size increases from 6 bits to 14 bits. The EbT throughput does not change with CtS configurations, hence the graph show only one data point for EbT.

From the figure we observe that when the frame size is 6 bits, the CtS throughput is highest because of the relatively low delay required to transmit the same amount of data.

As the frame size increases, the throughput decreases exponentially. This is expected as the result of the energy-throughput tradeoff we discussed in section 6.2. Note that when the CtS frame size is less than 8 bits, the total throughput of CtS is higher than that of EbT. This is a direct result of CtS frame multiplexing around the sink. However, as frame size increases, the delay between start and stop signals becomes dominant, and the aggregate throughput decreases accordingly.

Figure 30 (b) shows the energy consumed per data bit delivered for EbT and CtS. The energy consumption of CtS is about 80% of EbT energy consumption when the frame size is 6 bits, and is only 20% of EbT energy consumption when the frame size is 8 bits or higher. Interestingly, the energy consumption of CtS at 6 bit frame size is much higher compared to the energy for longer frame sizes. This is because, when the frame size is small, more CtS signals are transmitted, thereby causing more signal collisions and retransmissions. But when the frame size is 8 bits, very few collisions happen, thus the energy per bit is much lower. Beyond this frame size, the energy performance of CtS does not improve too much because at 8-bit frame size the collisions are already low enough.

This observation infers that it is important to find the optimal CtS frame size for a network topology. From the figure 30 (a) we see the throughput of CtS decreases exponentially as data frame size increases beyond 8 bits, but the energy improvement is negligible. Thus there is little reason to choose a longer frame size for this topology. At 8-bit frame size, the throughput of CtS is almost the same as the EbT throughput, while the energy per bit is only 20% that of EbT. This is an optimal point of operation we should use for this particular network topology.

7.4 Future Research

In the above CtS solutions we mainly focus on MAC layer and system solutions. Network-wide CtS solutions such as routing, flow control scheme are subjects of future research. In the following we discuss these aspects briefly.

- Routing solution

For CtS applications we assume relatively static sensor networks to reduce the necessities of control message exchanges. A routing algorithm computes the routes for all sensors to the sink at the network start up phase. After that these routes are propagated to every CtS node in the network. These routes remain static most of the time unless route failures occur due to lossy link or dead nodes. Under these circumstances a rerouting scheme is needed to find alternate routes.

Most reactive rerouting schemes in EbT communication involve broadcast of route requests and replies, hence are message-intensive. They cannot be applied in CtS, because of the limited resources available in a CtS environment, and the delay incurred when transmitting those messages using CtS scheme. A simple solution is to equip CtS nodes with data radios to enable the exchange of control messages quickly when necessary. The drawback is the complication of the node design for operations used only infrequently. It's also possible for the source or intermediate nodes to try to send CtS signals to other upstream nodes instead of the one leading to the dead route until a route to the sink is found. Compared to the previous solution, this solution is less expensive, but may not be as reliable. We will explore these design choices in our future research.

- Flow scheduling

In the CtS solution we discussed above, a CtS node cannot forward signals for different flows simultaneously, otherwise address conflicts may occur. The similar problem exists for EbT too as a result of many-to-one communication. But in EbT intermediate nodes switch between different flows on a packet basis, while for CtS the switch only occurs after a flow stops. To improve the fairness of CtS, we can reduce the duration of each flow such that the switch occurs more frequently. But what is the optimal granularity of a CtS flow is an interesting problem to study.

Two problems need to be solved to realize CtS flow scheduling: 1) how to suppress other flows at the bottleneck node and 2) how to resume another flow when the current flow stops. Flow control signals can be used to achieve the two purposes, but they increases the address requirements for each link. Fairness at the network level is an important aspect of a flow scheduling scheme. Consequently, we need to determine which flow to activate, and which flow to suppress and at what time. All the above factors should be taken into consideration when designing the scheduling solution.

- Downstream data delivery

  For the CtS solutions we only considered data delivery in the upstream direction (i.e. from sensors to sink). Although majority of data flow in this direction, downstream data flow is also indispensable for any sensor network. The sink needs to send control messages, software updates or synchronization information periodically to sensor nodes, and all these operations are accomplished by downstream data delivery. The problems needed to be solved in this context are quite different from those of the upstream direction.

  Several problems need to be solved if we are to use CtS in the downstream direction: 1)Addresses: In upstream data delivery CtS signals carry only local addresses to reduce overhead. This is because all signals eventually go to the sink, thus global addresses are necessary only in higher level packets. But in the downstream direction, each CtS signal has to carry at least the global address of the destination node for correct reception. 2) Routing: Routes from sensors to sink can be statically configured because they remain the same most of the time, and each node only need to remember it's next hop to the sink. But in the downstream direction a signal's destination can be anywhere in the network so the routes cannot be easily preconfigured. A possible solution is for the sink to compute the route to destination first and embed routing information in the CtS signal to be delivered. But how this can be achieved in an efficient way remains a problem to tackle. 3) Broadcast: Downstream data in sensor networks are usually broadcast messages. In this case, the above two problems are avoided. But other concerns such as avoiding broadcast storm, leveraging the broadcast nature of the wireless medium, are interesting problems to study. In the worst case we can always fall back to EbT communication for downstream data delivery. However, it is more desirable to have a sensor network relying purely on CtS communication, so that the hardware cost and energy consumption are minimized.

7.5 Conclusions

In this section we present a practical CtS solution that realize the basic strategy we proposed in section 6. Radio solutions, framing, addressing, counting and synchronization mechanisms are introduced to address the design challenges we discussed in Section 6.6. Simulation results prove that this solution enables significant performance improvement of CtS over EbT both in terms of energy and throughput.

CHAPTER VIII

CONCLUSIONS

In this dissertation we addressed the problem of energy efficient communication in wireless sensor networks. We approach this problem at two different levels: macroscopic and microscopic.

At the macroscopic level, we study the energy efficiency of correlation aware data aggregation trees under various sensor network conditions and the tradeoffs involved in using them. We show that the energy improvement of a correlation aware tree tends to be bounded by a small constant under many network scenarios. Furthermore, the improvement corresponds to when the additional cost of establishing a correlation aware structure is not taken into account, in the presence of which the improvement will be further reduced.

We also characterize the maximum useable delay bound for achieving the maximum energy efficient structure. We show that the maximum useable delay bound is a small constant times the delay along the maximum length shortest-path in the default shortest path tree.

For different correlation models, we find that the energy efficiency of aggregation aware tree increases with correlation degree, and an aggregation tree with the lowest delay bound is the most energy efficient structure for low to moderate correlation degrees. Higher delay bounds help improve aggregation efficiency only when correlation degree is sufficiently high.

Starting from the insights obtained from the above investigation, we propose a simple, scalable, and distributed aggregation tree that does not require any centralized coordination while still achieves potential cost benefits due to efficient aggregation. The SCT structure is instantaneously constructed during the course of a single query delivery and is a fixed structure that is efficient for wide range of node densities, source densities, source distributions and correlation degrees. The SCT approach, with its highly manageable structure, ensures low maintenance overhead of the aggregation structure, while also addressing many practical challenges related to data gathering in wireless sensor networks.

At the microscopic level, we explore a new communication strategy that is based on conveying information using silence as opposed to energy based transmissions. We formally define the concept of CtS, and identify its basic energy-delay trade-off as a communication strategy. To address the inherent low throughput of basic CtS strategy, we propose unique optimization strategies that either alleviate the energy-delay trade-off, or in some cases even improve the throughput performance when compared to the traditional EbT scheme. We also identify and discuss several research challenges that exist in realizing both the optimization strategies and traditional MAC layer functionalities.

Based on the previous investigation we present a solution suite that practically realizes the CtS strategy in wireless sensor networks. This solution encompasses hardware, physical layer and MAC layer design, including addressing, framing, reliability, sequencing contention resolution, and synchronization solutions. Optimization strategies appropriate for the context are also included in the suite. To evaluate the performance of this solution, we compare it with the EbT scheme under the same network configurations using NS2. Simulation results prove that this solution enables significant performance improvement of CtS over EbT both in terms of energy and throughput. To fully realize the potentials of CtS strategy, we also point out several future research directions in this area.

We believe the contribution of our work lies not only in the new strategies and principles we identified and the new solutions we proposed, but also in the new research directions we established through these discoveries. We believe future researches can be built upon our work to realize more efficient communication solutions for wireless sensor networks, as well as many other types of networks.

REFERENCES

[1] "Chipcon CC1000 datasheet," (Chipcon Technology).

[2] "MICA2 multi-sensor module (MTS300/310)," (Crossbow Technology Inc.).

[3] "PC4500/PC4800 developer's reference manual.," in *Document number: 710-004247*, (Aironet Wireless Communications).

[4] "IEEE 802.11 working group: the standards for wireless LANs," (www.ieee802.org), August 2004.

[5] "IEEE 802.15.4 WPAN-LR task group," (www.ieee802.org/15/pub/TG4.html), May 2005.

[6] "PicoRadio: Berkeley wireless research center," (www.bwrc.eecs.berkeley.edu), July 2005.

[7] Abbas El Gamal, Chandra Nair, B. P. E. U. B. and Zahedi, S., "Energy-efficient scheduling of packet transmissions over wireless networks," in *INFOCOM*, 2002.

[8] A.Goel and D.Estrin, "Simultaneous Optimization for Concave Cost: Single Sink Aggregation or Single Source Buy-at-Bulk," in *ACM-SIAM Symposium on Discrete Algorithm*, 2003.

[9] Akyildiz, I. F., Su, W., Sankarasubramaniam, Y., and Cayirci, E., "A Survey on Sensor Networks," in *IEEE Communications Magazine*, Aug. 2002.

[10] Amis, A. D., Prakash, R., Huynh, D., and Vuong, T., "Max-Min D-cluster Formation in Wireless Ad Hoc Networks," in *Proceedings of The 19th Conference on Comuter Communications(INFOCOM'00)*, (Tel-Aviv, Israel), pp. 32–41, 2000.

[11] Bandyopadhyay, S. and Coyle, E. J., "An Energy Efficient Hierarchical Clustering Algorithm for Wireless Sensor Networks," in *Proceedings of The 22th Conference on Comuter Communications(INFOCOM'03)*, (Sanfrancisco, CA), pp. 1713 – 1723, 2003.

[12] Bao, L. and Garcia-Luna-Aceves, J. J., "A new approach to channel access scheduling for ad hoc networks," in *MobiCom '01: Proceedings of the 7th annual international conference on Mobile computing and networking*, (New York, NY, USA), pp. 210–221, ACM Press, 2001.

[13] Bhaskar Krishnamachari, D. E. and Wicker, S. B., "The impact of data aggregation in wireless sensor networks," in *Proceedings of the 22nd International Conference on Distributed Computing Systems*, pp. 575–578, 2002.

[14] Chen, B., Jamieson, K., Balakrishnan, H., and Morris, R., "Span: An energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks," in *Proceedings of the 7th ACM International Conference on Mobile Computing and Networking*, (Rome, Italy), pp. 85–96, July 2001.

[15] Chunlong Guo, L. C. Z. and Rabaey, J., "Low power distributed mac for ad hoc sensor radio networks," in *Global Telecommunications Conference, 2001. GLOBECOM '01. IEEE*, pp. 2944–2948, 2001.

[16] Cristescu, R., Beferull-Lozano, B., and Vetterli, M., "On Network Correlated Data Gathering," in *INFOCOM*, (Hong Kong), Mar. 2004.

[17] Cristescu, R. and Vetterli, M., "Power Efficient Gathering of Correlated Data: Optimization,NP-Completeness and Heuristics," in *The Fourth ACM International Symposium on Mobile Ad Hoc Networking and Computing*, June 2003.

[18] Du, D.-Z. and Hwang, F. K., "An approach for proving lower bounds: Solution of gilbert-pollak's conjecture on steiner ratio," in *FOCS 1990*, pp. 76–85.

[19] E.D Kaluarachi, Z. G. and Wilson, B., "Digital pulse interval modulation for optical free space communication links," in *IEE Colloq. On Optical Free Space Communication Links*, Feb. 1996.

[20] Elson, J. and Rmer, K., "Wireless sensor networks: a new regime for time synchronization," *SIGCOMM Comput. Commun. Rev.*, vol. 33, no. 1, pp. 149–154, 2003.

[21] Eugene Shih, SeongHwan Cho, N. I. R. M. A. S. A. W. and Chandrakasan, A., "Physical layer driven protocol and algorithm design for energy-efficient wireless sensor networks," in *MOBICOM*, 2001.

[22] Gallager, R., "Basic limits on protocol information in data communication networks," in *IEEE Transaction on Information Theory*, pp. 385–398, July 1976.

[23] Gu, L. and Stankovic, J. A., "Radio-triggered wake-up for wireless sensor networks," *Real-Time Syst.*, vol. 29, no. 2-3, pp. 157–182, 2005.

[24] Gupta, A., Pal, M., Ravi, R., and Sinha, A., "Boosted Sampling: Approximation Algorithms for Stochastic Optimization," in *STOC*, 2004.

[25] Hashemi, H. Hajimiri, A., "Concurrent multiband low-noise amplifiers-theory, design, and applications," *Microwave Theory and Techniques, IEEE Transactions on*, vol. 50, pp. 288 – 301, 2002.

[26] Hill, J. and Culler, D., "A wireless embedded sensor architecture for system-level optimization," tech. rep., U.C. Berkeley, 2001.

[27] Intanagoniwawat, C., Estrin, D., Govindan, R., and Heidemann, J., "Impact of Network Density on Data Aggregation in Wireless Sensor Networks," in *International Conference on Distributed Computing Systems (ICDCS'02)*, (Vienna, Austria), July 2002.

[28] Intanagoniwawat, C., Govindan, R., and Estrin, D., "Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks," in *MOBICOM*, (Boston, MA, USA), Aug. 2000.

[29] Karger, D. R. and Minkoff, M., "Building Steiner Trees with Incomplete Global Knowledge," in *Proceedings of the 41th Annual IEEE Symposium on Foundations of Computer Science*, 2000.

[30] Karp, B. and Kung, H., "GPSR: Greedy Perimeter Stateless Routing for Wireless Networks," in *In Mobile Computing and Networking*, 2000.

[31] Kashani, Z.H.; Shiva, M., "Bch coding and multi-hop communication in wireless sensor networks," in *Wireless and Optical Communications Networks, 2006 IFIP International Conference on*, pp. 1 – 5, 2006.

[32] M.A. Youssef, M. Y. and Arisha, K., "A constrained shortest-path energy-aware routing algorithm for wireless sensor networks," in *WCNC*, March 2002.

[33] Mhatre, V. and Rosenberg, K., "Design Guidelines for Wireless Sensor Networks: Communication, Clustering and Aggregation," in *Elsevier Science Journal*, Aug. 2003.

[34] Miller, M. J. and Vaidya, N. H., "Improving the energy consumption of sensor network wake-up protocols," tech. rep., Department of Computer Science, University of Illinois at Urbana-Champaign, August 2004.

[35] Miller, M.J. Vaidya, N., "Minimizing energy consumption in sensor networks using a wakeup radio," in *Wireless Communications and Networking Conference, WCNC 2004)*, pp. 2335– 2340, 21-25 March 2004.

[36] Min, R., Bhardwaj, M., Cho, S.-H., Shih, E., Sinha, A., Wang, A., and Chandrakasan, A., "Low-power wireless sensor networks," in *VLSID '01: Proceedings of the The 14th International Conference on VLSI Design (VLSID '01)*, (Washington, DC, USA), p. 205, IEEE Computer Society, 2001.

[37] P. Agrawal, T. T. and Ananda, A., "A lightweight protocol for wireless sensor networks," in *WCNC*, march 2003.

[38] Pattem, S., K. B. and Govindan, R., "The impact of spatial correlation on routing with compression in wireless sensor networks," in *International Symposium on Information Processing in Sensor Networks*, pp. 28 – 35, April 2004.

[39] Prabhakar, B., Uysal-Biyikoglu, E., and Gamal, A. E., "Energy-efficient transmission over a wireless link via lazy packet scheduling," in *INFOCOM*, pp. 386–394, 2001.

[40] Ramanathan, S., "A unified framework and algorithm for channel assignment in wireless networks," *Wirel. Netw.*, vol. 5, no. 2, pp. 81–94, 1999.

[41] Ratnasamy, S., Karp, B., Yin, L., Yu, F., Estrin, D., Govindan, R., and Shenker, S., "GHT - A Geographic Hash-Table for Datacentric Storage," in *In First ACM International Workshop on Wireless Sensor Networks and their Applications*, 2002.

[42] Rhee, I., Warrier, A., and Xu, L., "Randomized dining philosophers to tdma scheduling in wireless sensor networks," tech. rep., Department of Computer Science, North Carolina State University, April 2005.

[43] Ruffieux, D., "A high-stability, ultra-low-power quartz differential oscillator circuit for demanding radio applications," in *Proceedings of the 28th European Solid-State Circuits Conference*, pp. 85– 88, September 2002.

[44] S.-J. Park, R. Vedantham, R. S. and I.F.Akyildiz, "A scalable approach for reliable downstream data delivery in wireless sensor networks," in *MOBIHOC*, 2004.

[45] Salama, H.F.; Reeves, D. V. Y., "Evaluation of multicast routing algorithms for real-time communication on high-speed networks," in *IEEE Journal on Selected Areas in Communications*, vol. 15, pp. 332 – 345, April 1997.

[46] SCHURGERS, C., A. O. and SRIVASTAVA, M. B., "Modulation scaling for energy aware communication systems," in *International Symposium on Low Power Electronics and Design (ISLPED)*, (Huntington Beach, California), 2001.

[47] Schurgers, C., Tsiatsis, V., Ganeriwal, S., and Srivastava, M., "Optimizing sensor networks in the energy-latency-density design space," *IEEE Transactions on Mobile Computing*, vol. 01, no. 1, pp. 70–80, 2002.

[48] Singh, S. and Raghavendra, C. S., "Pamaspower aware multi-access protocol with signalling for ad hoc networks," *SIGCOMM Comput. Commun. Rev.*, vol. 28, no. 3, pp. 5–26, 1998.

[49] Takahashi, H. and Matsuyama, A., "An Approximate Solution for the Steiner Tree Problem in Graphs," in *Math. Japonica 24*, vol. 6, pp. 573–577, 1980.

[50] Tanenbaum, A. S., *Computer Networks; 3rd edition.* Prentice Hall, 1996. TAN a 96:1 P-Ex.

[51] Thad Starner, Steve Mann, B. R. J. L. J. H. D. K. R. W. P. and Pentland, A., "Augmented reality through wearable computing," vol. 6, (Presence, Special Issue on Augmented Reality), 1997.

[52] V. Raghunathan, S. Ganeriwal, C. S. and Srivastava, M. B., "e2wfq: An energy efficient fair scheduling policy for wireless systems," in *in International Symposium on Low Power Electronics and Design (ISLPED02)*, 2002.

[53] W. Yuan, S. Krishnamurthy, S. K. T., "Synchronization of multiple levels of data fusion in wireless sensor networks," in *IEEE Global Communications Conference (GLOBECOM'03)*, December 2003.

[54] Wang, N.-Y. B. and Chang, R.-C., "An Upper Bound for the Average Length of the Euclidean Minimum Spanning Tree," in *J. Computer Math*, vol. 30, pp. 1–12, 1989.

[55] Wei Ye, John Heidemann, D. E., "An energy-efficient mac protocol for wireless sensor networks," in *INFOCOM*, 2002.

[56] Wendi R. Heinzelman, A. C. and Balakrishnan, H., "Energy-efficient communication protocol for wireless microsensor networks," in *Proceedings of the 33rd Hawaii International Conference on System Sciences*, 2000.

[57] Y. Zhu, K. S. and Sivakumar, R., "Practical limits on achievable energy improvements and useable delay tolerance in correlation aware data gathering in wireless sensor networks," in *IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Network (SECON)*, (Santa Clara, California), 2005.

[58] Y. Zhu, R. Vedantham, S.-J. P. and Sivakumar, R., "A scalable correlation aware aggregation strategy for wireless sensor networks," in *IEEE International Conference on Wireless Internet (WICON)*, (Budapest, Hungary), 2005.

[59] Yang Yu, B. K. and Prasanna, V. K., "Energy-latency tradeoffs for data gathering in wireless sensor networks," in *INFOCOM*, 2004.

[60] Younis, O. and Fahmy, S., "HEED: A Hybrid, Energy-efficient, Distributed Clustering Approach for Ad-hoc Sensor Networks," *IEEE Transactions on Communications*, vol. 3, no. 4, pp. 366–379, 2004.

[61] Yu, Y. and Prasanna, V. K., "Energy-balanced multi-hop packet transmission in wireless sensor networks," in *GLOBECOM*, 2003.

[62] Yuan, W., Krishnamurthy, S. V., and Tripathi, S. K., "Synchronization of Multiple Levels of Data Fusion in Wireless Sensor Networks," in *IEEE GLOBECOM*, 2003.

[63] Zhu, Y. and Sivakumar, R., "Communication through silence in wireless sensor networks," in *ACM International Conference on Mobile Computing and Networking (MOBICOM)*, (Cologne, Germany), 2005.

[64] Zorzi, M. Rao, R., "Coding tradeoffs for reduced energy consumption in sensor networks," in *Personal, Indoor and Mobile Radio Communications, 2004. PIMRC 2004. 15th IEEE International Symposium on*, pp. 206– 210, 2004.

VITA

Yujie Zhu was born in Sichuan Province, China. She received her B.E. degree in Electrical Engineering from Beijing University of Aeronautics and Astronautics in 1994, and her M.E. degree from the same school in 1997. After that she worked as a system engineer in the Guangzhou Data Network Management Center of China Telecom from 1997 to 2001. She joined the PhD program of School of Electrical and Computer Engineering at Georgia Tech in 2001, and has been a Research Assistant in Dr. Sivakumar's GNAN research group.