Application-Specific Protocol Architectures for Wireless Networks

by

Wendi Beth Heinzelman

B.S., Cornell University (1995)
M.S., Massachusetts Institute of Technology (1997)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2000

© Wendi Beth Heinzelman, MM. All rights reserved.

The author hereby grants to MIT permission to reproduce and distribute publicly paper
and electronic copies of this thesis document in whole or in part.

Author ....
Department of Electrical Engineering and Computer Science
May 19, 2000

Certified by ....
Anantha P. Chandrakasan
Associate Professor
Thesis Supervisor

Certified by ....
Hari Balakrishnan
Assistant Professor
Thesis Supervisor

Accepted by ....
Arthur C. Smith
Chairman, Department Committee on Graduate Students

Application-Specific Protocol Architectures for Wireless Networks

by

Wendi Beth Heinzelman

Submitted to the Department of Electrical Engineering and Computer Science
on May 19, 2000, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Abstract

In recent years, advances in energy-efficient design and wireless technologies have enabled exciting new applications for wireless devices. These applications span a wide range, including real-time and streaming video and audio delivery, remote monitoring using networked microsensors, personal medical monitoring, and home networking of everyday appliances. While these applications require high performance from the network, they suffer from resource constraints that do not appear in more traditional wired computing environments. In particular, wireless spectrum is scarce, often limiting the bandwidth available to applications and making the channel error-prone, and the nodes are battery-operated, often limiting available energy.

My thesis is that this harsh environment with severe resource constraints requires an application-specific protocol architecture, rather than the traditional layered approach, to obtain the best possible performance. This dissertation supports this claim using detailed case studies on microsensor networks and wireless video delivery. The first study develops LEACH (Low-Energy Adaptive Clustering Hierarchy), an architecture for remote microsensor networks that combines the ideas of energy-efficient cluster-based routing and media access together with application-specific data aggregation to achieve good performance in terms of system lifetime, latency, and application-perceived quality. This approach improves system lifetime by an order of magnitude compared to general-purpose approaches when the node energy is limited. The second study develops an unequal error protection scheme for MPEG-4 compressed video delivery that adapts the level of protection applied to portions of a packet to the degree of importance of the corresponding bits. This approach obtains better application-perceived performance than current approaches for the same amount of transmission bandwidth. These two systems show that application-specific protocol architectures achieve the energy and latency efficiency and error robustness needed for wireless networks.

Thesis Supervisor: Anantha P. Chandrakasan
Title: Associate Professor

Thesis Supervisor: Hari Balakrishnan
Title: Assistant Professor

This thesis is dedicated to the memory of

Robert H. Rabiner

1942-1997

"The first Rabiner at the 'Tute"

Acknowledgments

I would like to begin by thanking Professor Anantha Chandrakasan, my thesis advisor and mentor for the past 5 years. Anantha has been a wonderful advisor, providing me with support, encouragement, and an endless source of ideas. His breadth of knowledge and his enthusiasm for research amazes and inspires me. I thank him for the countless hours he has spent with me, discussing everything from research to career choices, reading my papers, and critiquing my talks. His assistance during my time at MIT has been invaluable—my life has been enriched professionally, intellectually, and personally by working with Anantha.

I would also like to thank Professor Hari Balakrishnan, my Ph.D. co-advisor. Hari has been a great advisor. His enthusiasm for research and his vision for the future have been an inspiration. It was a wonderful learning experience to watch Hari build a successful research program in such a short time. Hari has given me support and encouragement, and his advice and feedback about my research have greatly enhanced and strengthened the work. I thank him for all the time and energy he has invested into my research.

I would like to thank Dr. Gary Shaw for his initial feedback on the general area of sensor networks and for reading and commenting on my thesis. His suggestions were very helpful. I would also like to thank Professor John Guttag for valuable discussions about research and career opportunities. There are many good teachers at MIT, but I was fortunate enough to work with two outstanding ones—Professor Alan Oppenheim and Professor Gregory Wornell. I thank them for showing me what it takes to be a great teacher and for all their advice through the years.

I am grateful to Dr. Raj Talluri, Dr. Madhukar Budagavi and Dr. Jennifer Webb at Texas Instruments for helping start my Ph.D. research with exciting ideas about transmitting video over wireless networks. It was a great experience to work with them.

While at MIT, I had the privilege of interacting with wonderful, bright, and talented people. They have taught me much, and their advice, feedback, and friendship have made my Ph.D. experience both more educational and more fun: Raj Amirthirajah, Manish Bhardwaj, SeongHwan Cho, Travis Furrer, Jim Goodman, Vadim Gutnik, James Kao, Rex Min, Jose Oscar Mur-Miran, Chee We Ng, Eugene Shih, Tom Simon, Amit Sinha, Paul-Peter Sotiriadis, Zoe Teegarden, Alice Wang, Duke Xanthopoulos, and all the members of the NMS group. It has been a pleasure to work with all of you. I would especially like to thank Alice Wang for all her help with beamforming algorithms for data aggregation—I've enjoyed our collaborations! I also wish to thank Margaret

Flaherty for all her help with administrative matters.

I am grateful to the Eastman Kodak Company for supporting my research through a Kodak fellowship. I hope to continue my close ties with them in the future.

I would like to thank my family, my sisters Sheri Rabiner Gordon and Joni Rabiner, my grandmother Gloria Rabiner, my brother-in-law Paul Gordon, my sister-in-law Cathy Heinzelman, and my in-laws David and Columba Heinzelman, for all their love and support. I am very lucky to have such wonderful family members. I have enjoyed sharing my MIT experiences with all of you!

I am indebted to my parents, Suzanne and Lawrence Rabiner, for everything that they have given to me. They taught me the value of knowledge, the joy of love, and the importance of family. They have stood by me in everything I have done, providing constant support, encouragement, and love. They are an inspiration to me in all that they do.

Finally, I would like to thank my wonderful husband, Stephen Heinzelman. Steve has been my best friend, my “true companion” since the day I met him almost 9 years ago. He has shown me how to enjoy life to the fullest, from exotic adventures traveling the world to time spent relaxing together at home. Steve has kept me happy and sane during the Ph.D. process, and I thank him for all his patience and his never-ending optimism when I came home late, frustrated, and stressed. He keeps my life filled with laughter and love, and I know that there is nothing I cannot accomplish with Steve by my side.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In recent years, we have seen a proliferation of wireless devices, including cellular phones, pagers, laptops, and personal digital assistants (PDAs). In fact, it is predicted that wireless data access will exceed wireline access by the year 2004 (see Figure 1-1) [98]. Advances in energy-efficient design and wireless technologies have enabled portable devices to support several important wireless applications, including real-time multimedia communication [77], medical applications, surveillance using microsensor networks [5, 17, 19, 25], and home networking applications [12, 43].

An important challenge in the design of wireless and mobile systems is that two key resources—communication bandwidth and energy—are significantly more limited than in a tethered network environment. These restrictions require innovative communication techniques to increase the amount of bandwidth per user and innovative design techniques and protocols to use available energy efficiently. Furthermore, wireless channels are inherently error-prone and their time-varying characteristics make it hard to consistently obtain good performance. Communication protocols must be designed to adapt to current conditions instead of being designed for worst-case conditions. Applications differ in which features are most important. For example, an application that supports wireless data communication might prefer longer latency in exchange for longer node lifetime. On the other hand, long latency is unacceptable for a cellular phone application. Similarly, lossy compression is unacceptable for data transfers, but represents a good trade-off to extend node lifetime for voice transfers. These unique considerations for different applications, coupled with the tight resource constraints of wireless systems, suggest the need for application-specific protocols.

My thesis is that application-specific protocol architectures must be designed and deployed in order to obtain the most efficient systems that achieve high performance and energy-efficiency in

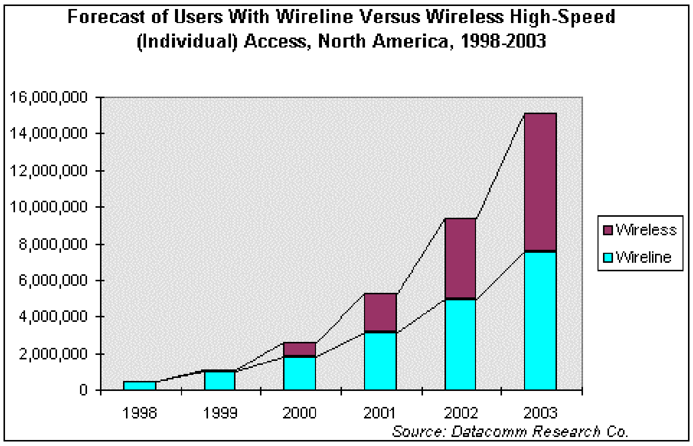


Figure 1-1: Predictions are that wireless data access will exceed wireline access by the year 2004 [reported by Datacomm Research Co., reprinted with permission from Wirelesstoday.com].

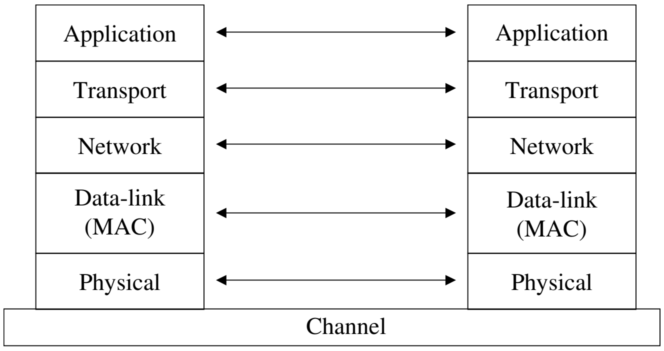


Figure 1-2: This dissertation focuses on protocol architectures that exploit application-specific information in the transport, network, and data-link layers of this protocol stack for specific wireless applications.

a wireless environment. Conventional network architectures are designed according to a layering approach, such as the one shown in Figure 1-2 [103]. Here, each layer of the system is designed separately and is independent of the application. A layered approach allows the system design to be broken into smaller pieces that can be developed independently. Protocols designed using such an approach, while reusable by many different applications, are not optimal for any given application. Rather than using a general-purpose protocol architecture, we make the case that systems will be more efficient if they are designed to exploit features of the applications they are supporting. A cross-layer architecture that exploits features of the application can achieve greater performance than general-purpose protocols.

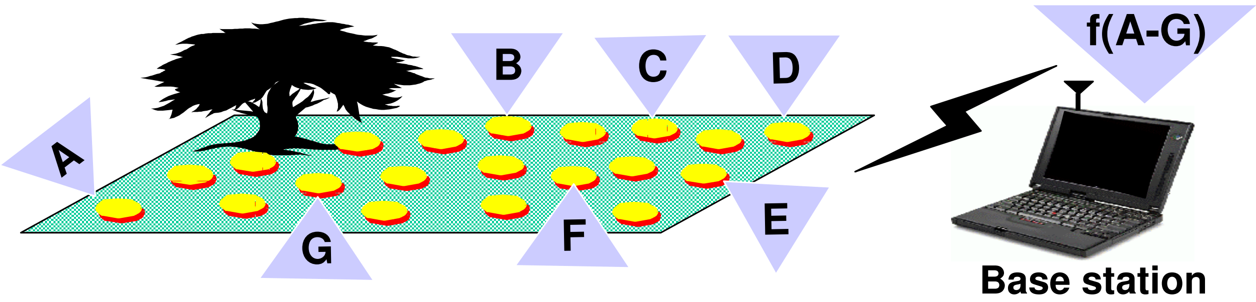


Figure 1-3: A wireless microsensor network. Each node obtains a certain view of the environment. By intelligently combining the views of the nodes, the end-user can remotely monitor events in the environment.

The use of cross-layer design optimizations has been demonstrated in the context of wireless Internet delivery [8], application-controlled routing [3, 41, 48], wireless multimedia delivery [46, 49], and protocol frameworks for active wireless networks [53]. This dissertation enhances our understanding of cross-layer design by developing and analyzing application-specific protocol architectures for two new situations—wireless microsensor networks and the delivery of real-time video over wireless networks. We show that by using cross-layer design optimizations, our protocol architectures achieve the high performance and energy efficiency needed under the tight constraints of a tetherless network.

1.1 Wireless Microsensor Networks

A sensor is any device that maps a physical quantity from the environment to a quantitative measurement. Advances in sensor technology, low-power analog and digital electronics, and low-power radio frequency (RF) design have enabled the development of small, relatively inexpensive and low-power sensors, called *microsensors*. Microsensors are equipped with a sensor module (e.g., acoustic, seismic, image sensor) capable of sensing some quantity about the environment, a digital processor for processing the signals from the sensor and performing network protocol functions, a radio module for communication, and a battery to provide energy for operation. Each sensor obtains a certain "view" of the environment, as shown in Figure 1-3. A given sensor's view of the environment is limited both in range and in accuracy; it can only cover a limited physical area of the environment and, depending on the quality of the hardware, may produce noisy data. Combining or *aggregating* the views of the individual nodes allows users to accurately and reliably monitor an environment. To enable remote monitoring of an environment, the nodes must send

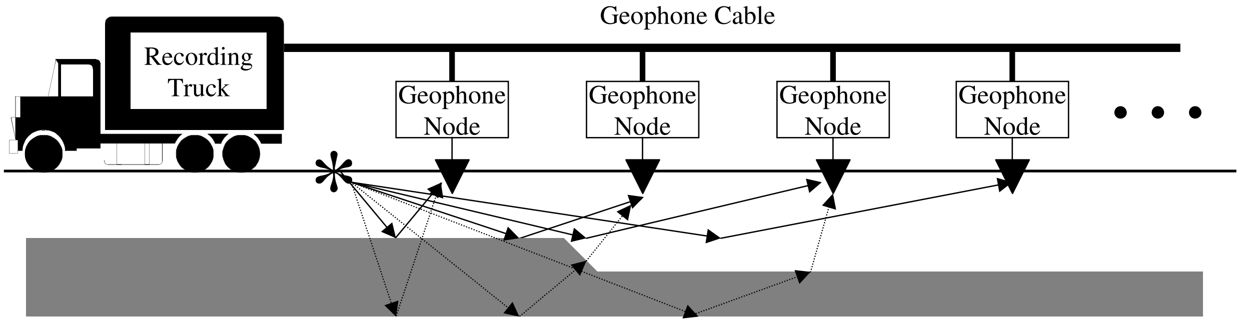


Figure 1-4: The old paradigm for extracting data from the environment included the use of large, expensive, bulky macrosensor nodes connected via a tethered network to the end-user. This figure shows a picture of seismic exploration.

the high-level description of events to a distant base station, through which an end-user can access the information.

Wireless microsensor networks represent a new paradigm for extracting data from the environment. Conventional systems use large, expensive macrosensors that are often wired directly to an end-user and need to be accurately placed to obtain the data. For example, the oil industry uses large arrays of geophone sensors attached to huge cables to perform seismic exploration for oil [47, 80], as shown in Figure 1-4. These sensor nodes are very expensive and require large amounts of energy for operation. The sensors must be placed in exact locations, since there are a limited number of nodes extracting information from the environment. Furthermore, deployment of these nodes and cables is costly and awkward, requiring helicopters to transport the system and bulldozers to ensure the sensors can be placed in exact positions. There would be large economic and environmental gains if these large, bulky, expensive macrosensor nodes could be replaced with hundreds of cheap microsensor nodes that can be easily deployed. This would save significant costs in the nodes themselves as well as in the deployment of these nodes. These microsensor networks would be fault-tolerant, as their sheer number of nodes can ensure that there is enough redundancy in data acquisition that not all nodes need to be functional. Using wireless communication between the nodes would eliminate the need for a fixed infrastructure.

Wireless microsensor networks enable the reliable monitoring of a variety of environments for applications that include home security, machine failure diagnosis, chemical/biological detection, medical monitoring, and surveillance. Deploying hundreds or thousands of nodes in a wireless microsensor network brings new benefits to these sensing applications, including:

- Extended range of sensing. Single macrosensor nodes can only extract data about events in a limited physical range. In contrast, microsensor networks enable large numbers of nodes to be physically separated; while nodes located close to each other will have correlated data (e.g., these nodes will be gathering data about the same event), nodes that are farther apart will be able to extract information about different events.

- Fault-tolerance. Ensuring that several nodes are located close to each other and hence have correlated data makes these systems much more fault tolerant than single macrosensor systems. If the macrosensor node fails, the system cannot function, whereas if a small number of microsensor nodes from a network fail, there is enough redundancy in the data from different nodes that the system may still produce acceptable quality information.

- Improved accuracy. While an individual microsensor's data might be less accurate than a macrosensor's data, combining the data from nodes increases the accuracy of the sensed data. Since nodes located close to each other are gathering information about the same event, aggregating their data enhances the common signal and reduces the uncorrelated noise.

- Lower cost. Even though there are many microsensors replacing each macrosensor, due to reduced size, reliability, and accuracy constraints on microsensor nodes, these nodes are much cheaper than their macrosensor counterparts. Therefore, microsensor systems are less expensive than macrosensor systems.

In order to achieve these benefits, we must design protocols that enable microsensor networks to provide the necessary support to the sensing applications.

1.1.1 Design Goals for Wireless Microsensor Network Protocols

In order to design good protocols for wireless microsensor networks, it is important to understand the parameters that are important to the sensor applications. While there are many ways in which protocols are beneficial to the application, we use the following metrics:

- **Ease of deployment**. Sensor networks may contain hundreds or thousands of nodes, and they may need to be deployed in remote or dangerous environments. If these nodes are small enough and cheap enough, we can imagine throwing hundreds or thousands of microsensors

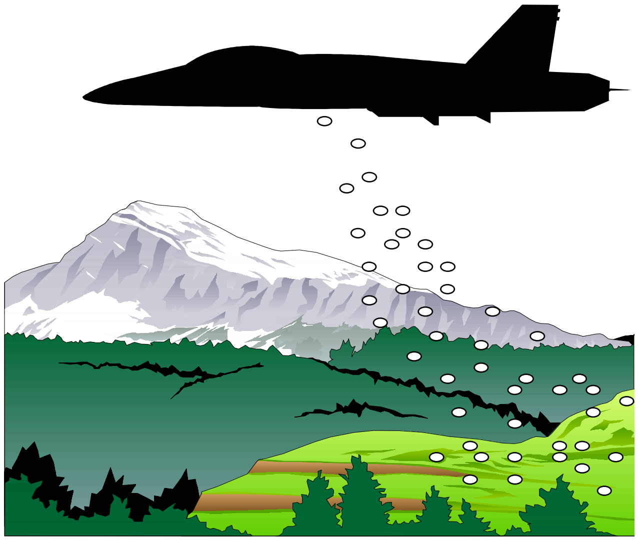


Figure 1-5: Microsensor nodes can be dropped from planes to enable monitoring of remote or dangerous areas. This requires self-configuring protocols that do not rely on a fixed infrastructure.

from a plane flying over a remote or dangerous area to allow us to extract information in ways that would not have been possible otherwise (see Figure 1-5).

- **System lifetime**. These networks should function as long as possible. System lifetime can be measured using generic parameters, such as the time until the nodes die, or it can be measured using application-specific parameters, such as the time until the sensor network is no longer providing acceptable quality results (e.g., there are too many missed events).

- **Latency**. Data from sensor networks are typically time-sensitive, so it is important to receive the data in a timely manner. Long delays due to processing or communication may be unacceptable.

- **Quality**. This parameter measures the accuracy with which the result of the sensor network matches what is actually occurring in the environment. Although this is an application-specific and data-dependent quantity, one possible application-independent method of determining quality is to determine the amount of data (either actual or aggregate) received at the base station. The more data the base station receives, the more accurate its view of the remote environment will be.

Tradeoffs can be made among these different parameters, and algorithms can be created that are scalable and adaptive to change the relative importance of the different parameters. For example, when energy is plentiful, the end-user may desire high-quality results. As the energy gets depleted, the end-user may request that the quality of the results be reduced in order to reduce the energy dissipation in the nodes and hence lengthen the total system lifetime. Thus microsensor network algorithms and protocols should be *power aware* such that energy usage is scaled appropriately for a given quality specification [10, 42].

1.1.2 Challenge: Meeting the Design Goals

As discussed in the previous section, it is important that microsensor networks be easily deployable, possibly in remote or dangerous environments. This requires that the nodes be able to communicate with each other even in the absence of an established network infrastructure. In addition, there are no guarantees about the locations of the sensors, such as the uniformity of placement. To function in such ad-hoc settings, microsensor networks should be *self-configuring*, requiring no global control to set-up or maintain the network.

In the scenario where the sensors are operating in remote or dangerous territory, it may be impossible to retrieve the nodes in order to recharge batteries. In other sensor network scenarios, such as machine monitoring or medical monitoring, it may just be inconvenient to replace the battery of a node. Therefore, the network should be considered to have a certain lifetime during which nodes have energy and can gather, process, and transmit information. This means that all aspects of the node, from the sensor module to the hardware and protocols, must be designed to be extremely energy-efficient. Decreasing energy usage by a factor of two can double system lifetime, resulting in a large increase in the overall usefulness of the system. In addition to reducing energy dissipation, protocols should be robust to node failures in order to maximize system lifetime. The protocols should be fault-tolerant, such that the loss of a small number of nodes does not greatly affect the overall system performance. In addition, the protocols should be scalable such that the addition of new nodes requires low overhead to incorporate the nodes into the existing network.

Events occurring in the environment being sensed may be time-sensitive. Therefore, it is often important to bound the end-to-end latency of data dissemination. Protocols should therefore minimize overhead and extraneous data transfers.

Researchers have been studying wireless networks for a number of years and have developed fairly sophisticated protocols for voice delivery using cellular networks and data delivery over wireless local area networks (WLANs) and ad-hoc data networks [14, 32, 69]. In cellular networks, nodes are organized into clusters where each node is able to communicate directly with the cluster base station. This requires a fixed infrastructure (placement of base stations) so that nodes can be connected to the network wherever they are. WLANs usually require point-to-point connectivity so any user can communicate with any other user, often without the use of a central base-station; these networks typically use some form of multi-hop routing. While these protocols are good at optimizing delay and fairness parameters, they are designed for applications where each user is creating data that may be transferred to any other user at any given time. These goals are very different than those of a wireless microsensor network. In a microsensor network, data sensed by each node are required at a remote base station, rather than at other nodes, and the data are being extracted from the environment, leading to large amounts of correlation among data signals. Therefore, the notion of "quality" in a microsensor network is very different than in a WLAN or cellular network. For sensor networks, the end-user does not require all the data in the network because (1) the data from neighboring nodes are highly correlated, making the data redundant, and (2) the end-user cares about a higher-level description of events occurring in the environment the nodes are monitoring. The quality of the network is therefore based on the quality of the aggregate data set, rather than the quality of the individual data signals; protocols should be designed to optimize for the unique, application-specific quality of a sensor network.

To summarize, wireless microsensor network protocols should be:

- self-configuring, to enable ease of deployment of the networks,
- energy-efficient and robust, to extend system lifetime,
- latency-aware, to get the information to the end-user as quickly as possible, and
- cognitive of the application-specific nature of sensor network quality.

The research presented in this dissertation on microsensor networks focuses on ways in which this last feature may be exploited to create a protocol architecture that optimizes the different desired features of these networks. This is accomplished by using application-level information in the design of all layers of the traditional protocol stack of Figure 1-2.

1.1.3 Solution: The LEACH Protocol Architecture

We have designed and implemented LEACH (Low-Energy Adaptive Clustering Hierarchy), a protocol architecture for wireless microsensor networks that achieves low energy dissipation and latency without sacrificing application-specific quality. Since data are correlated and the end-user only requires a high-level description of the events occurring in the environment the nodes are sensing, the nodes can collaborate locally to reduce the data that need to be transmitted to the end-user. Correlation is strongest among data signals from nodes that are close to each other, suggesting the use of a clustering infrastructure that allows nodes that are close to share data. Therefore, LEACH uses a clustering architecture, where the nodes in the cluster send their data to a local *cluster-head*. This node is responsible for receiving all the data from nodes within the cluster and aggregating this data into a smaller set of information that describes the events the nodes are sensing. Thus the cluster-head node takes a number of data signals and reduces the *actual* data (total number of bits) while maintaining the *effective* data (information content). The cluster-head node must then send the aggregate data set to the end-user.

Since there may be no fixed infrastructure with a high-energy node that can act as a cluster-head, one of the sensor nodes must take on this role. If this position was fixed, the cluster-head would quickly use up its limited energy and die, ending the communication ability of the rest of the nodes in the cluster as well. Therefore, LEACH includes rotation of this cluster-head position among all the nodes in the network to evenly distribute the energy load. In order to rotate cluster-head nodes and associated clusters, the cluster formation algorithm must ensure minimum overhead, in terms of time and energy.

Once clusters have been formed, the nodes must communicate their data to the cluster-head node in an energy-efficient manner. In LEACH, this is accomplished using a time-division multiple access (TDMA) protocol, which allows the nodes to shut down some internal components and enter a sleep state when they are not transmitting data to the cluster-head. In addition, using a TDMA approach in the steady-state ensures there are no collisions of data within the cluster, saving energy and time.

LEACH therefore uses the following techniques to exploit the application-specific functionality of a sensor network and achieve energy and latency efficiency: (i) randomized, adaptive, self-configuring cluster formation, (ii) localized control for data transfers, (iii) low-energy media access, and (iv) application-specific data processing, such as data aggregation or compression. The cluster formation algorithm allows each node to make autonomous decisions that result in good clusters being formed. This algorithm also minimizes the energy and latency for cluster formation, in order to minimize overhead to the protocol. Using local control to set up a TDMA schedule and implementing low-energy media access reduce energy dissipation and latency. Finally, local data processing achieves a large energy reduction by performing computation on the correlated data to greatly reduce the amount of data that must be transmitted long distances.

We perform an in-depth comparison of LEACH with a general-purpose routing architecture applied to wireless microsensor networks. This is done using an extension of the `ns` network simulator using wireless channel models and models for energy dissipation. Our simulation results show that LEACH achieves an order of magnitude increase in system lifetime compared to general-purpose approaches. Furthermore, LEACH reduces overall latency by an order of magnitude for a given quality.

1.2 Wireless Video Transmission

While microsensor networks present new challenges in the design of protocol architectures, more traditional applications can also benefit from protocols that use a cross-layer design. In particular, wireless transport of multimedia data (e.g., voice, video) presents several challenges due to the real-time requirements of the applications and the low bandwidth and error-prone nature of the wireless channel. These applications of wireless channels are becoming increasingly important. Figure 1-6 shows that there are expected to be seven hundred million subscribers to cellular services by the year 2003 [99]. The introduction of the 3^{rd} generation (3G) mobile systems based on the Global System for Mobile Communications (GSM) standard (being developed under a project called the "Third-Generation Partnership Project", or 3GPP [1]) will offer each user rates between 144 and 384 kbps [16], enough bandwidth to support real-time video in addition to audio and data services. It is predicted that after its inception, there will be over 50 million subscribers to 3G services, with that number increasing six-fold by the year 2005 (see Figure 1-7) [100].

Video cellular phones, such as the one shown in Figure 1-8, will enhance personal communications, entertainment, and web browsing. However, several technological barriers need to be overcome before wireless multimedia devices become commonplace. The hardware must support the high bitrates, memory, and data transfers required for video processing in an energy-efficient manner so as not to drain the portable device's battery. The compression algorithms need to minimize compression artifacts, such as blocking found in discrete cosine transform (DCT)-based coders and jerkiness associated with low frame rates. Finally, the channel coding algorithms need to reduce the raw channel bit error rate to an acceptable level.

While there is a need to produce high-quality video under the low bitrate and high error constraints of the wireless channel, there is a tradeoff in how many bits are allocated to the video

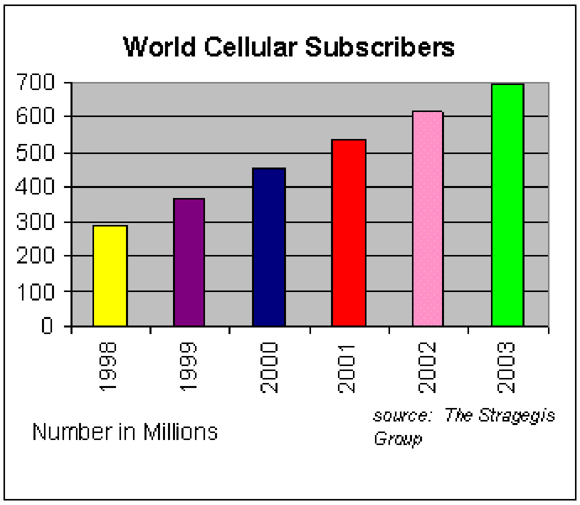


Figure 1-6: Prediction for the number of subscribers to cellular services [reported by The Stragegis Group, reprinted with permission from Wirelesstoday.com].

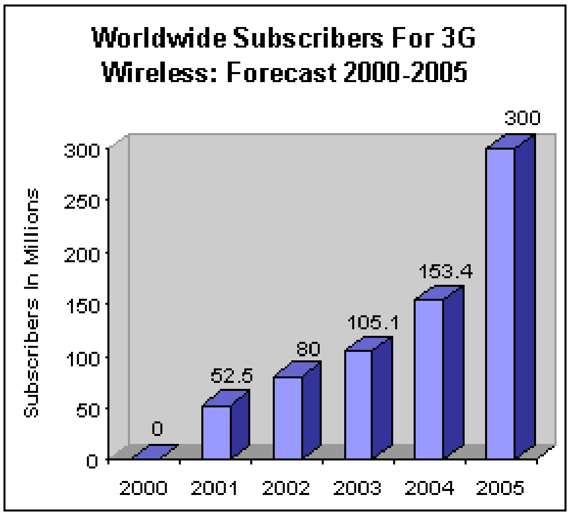


Figure 1-7: Prediction for the number of subscribers to 3G services [reported by Cahners In-Stat Group, reprinted with permission from Wirelesstoday.com].

Figure 1-8: 3^{rd} generation cellular standards will allow each user enough bandwidth to support multimedia applications. Video cellular phones will enhance communications, entertainment, and web browsing.

compression versus how many are allocated to the channel coding. The user wants to achieve the highest quality possible given a limited bandwidth and (possibly) high bit error rates. The challenge thus becomes how to make best use of the overhead bits so as to achieve the highest reconstructed video quality.

1.2.1 Challenge: Ensuring High-Quality Video Over Wireless Channels

Mobile multimedia terminals must be able to compress video for transmission over the low-bandwidth, error-prone wireless channel such that the decoder obtains high-quality reconstructed video. The standard video compression algorithms (e.g., H.263 [26], MPEG [56]) use predictive coding (where the difference between the current frame and a previous frame is encoded) and variable-length codewords to obtain a large amount of compression. This makes the compressed video bitstream sensitive to channel errors, as predictive coding causes errors in the reconstructed video to propagate in time to future frames of video, and the variable-length codewords cause the decoder to easily lose synchronization with the encoder in the presence of bit errors. This results in an unacceptably low quality of the reconstructed video, as depicted in Figure 1-9.

To make the compressed bitstream more robust to channel errors, the MPEG-4 video com-

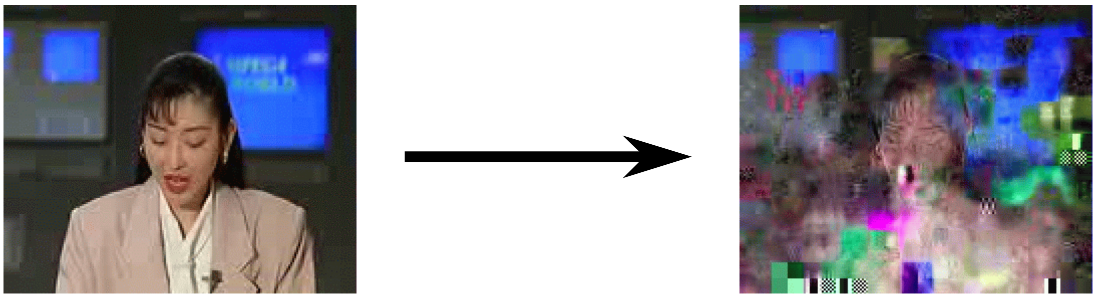


Figure 1-9: (a) Reconstructed image when there are no channel errors. (b) Reconstructed image when there are channel errors. Error propagation and loss of synchronization between the encoder and the decoder corrupt the reconstructed image.

pression standard incorporates several error resilience tools to enable detection, containment, and concealment of errors. These tools include resynchronization markers, header extension codes, data partitioning, and reversible variable-length coding [90, 89]. These techniques are effective when the bit error rate is less than about 10^{-3}. However, wireless channels can have much higher bit error rates [92]. In this case, a link-layer protocol that adds channel coding or forward error correction (FEC) can be used to bring the aggregate bit error rate down to less than 10^{-3}. The research presented in this dissertation on wireless multimedia networks focuses on ways in which application-level information can be exploited to create a link-layer protocol that achieves better performance than a general-purpose protocol for this application.

1.2.2 Solution: Unequal Error Protection

Application-specific information can be incorporated into the link-layer protocol for a wireless video system to improve reconstructed video quality for a given bandwidth. This dissertation describes our research in designing a link-layer protocol that exploits the inherent structure of an MPEG-4 compressed bitstream. This protocol uses *unequal error protection* to adjust the amount of protection to the level of importance of the corresponding bits [39, 102]. This ensures that there are fewer errors in the important portions of the bitstream. While an unequal error protection scheme that uses the same number of overhead bits as an equal error protection scheme actually leaves more errors in the channel decoded bitstream, these errors are in less important portions of the video bitstream. Therefore, an unequal error protection scheme will provide better *application-perceived* *quality* than an equal error protection scheme. Simulations show that unequal error protection improves reconstructed video quality by as much as 1 dB (with considerable visual improvement) compared to equal error protection that does not incorporate this application-specific information.

1.3 Contributions of this Dissertation

The results of our research show that designing application-specific protocol architectures provide the high performance needed under the tight constraints of the wireless channel. While a layering approach to protocol design allows functions to be designed and reasoned about independently, this approach will not guarantee high performance from the network. Rather than collapsing the protocol stack entirely, we found that using cross-layer design, whereby layers are exposed to information from other layers, produces the high performance required. In order to design such cross-layer systems, it is important to define the function and goals of the application and the relative importance of the different system parameters. Once these are known, appropriate trade-offs can be made to design protocol architectures that provide maximum benefit to the application.

For microsensor networks, the unique considerations are the function of the application, the need for ease of deployment, and the severe energy constraints of the nodes. These features led us to design a system where

- computation is performed locally to reduce the data set,
- network configuration is done without centralized control,
- received signal strength is used as a measure of the required transmit power, and
- media access control (MAC) and routing protocols enable nodes to remain in the sleep state for substantial periods, thereby reducing energy consumption.

Though these ideas were developed in the context of microsensor networks, the results can be applied to other types of wireless network protocol architectures. For example, wireless data and multimedia networks could benefit from designing MAC and routing that enable the nodes to go into the sleep state for long periods of time, as powering down portions of the node can save considerable energy. Home networking applications may benefit from the use of received signal strength rather than distance as a measure of how much energy would be required for communication between two nodes, as walls and furniture in the home may severely degrade the communication channel between two devices that are located close to each other. This enables power control to better reduce interference and energy dissipation of the device. Ad-hoc wireless networks can benefit from using self-configuring protocols to reduce management overhead. Finally, local data processing could be used in data networks. For example, in a network where a user broadcasts a request and receives multiple copies of the response, rather than forwarding all data to the end-user, intermediate nodes could remove redundant data to save communication and energy resources.

In our research with wireless video transport, we found that the two main considerations were the cost, in terms of energy required to send the bits and the latency in data transfer, and the performance of the system, in terms of reconstructed video quality. Thus it is important for a multimedia transport protocol to obtain maximum quality for a fixed cost or minimum cost for a fixed quality. This led us to design a wireless video link-layer protocol where

- the amount of channel protection is matched to the level of importance of the bits being protected and
- the overhead associated with producing different levels of protection is minimized.

Once again, these ideas can be extended to other types of networks. Most compression algorithms produce bitstreams with structure, which can be exploited using unequal error protection.

1.4 Dissertation Structure

This dissertation begins with a detailed discussion of general-purpose protocol architectures for wireless networks, including link-layer, media access control (MAC), and routing protocols (Chapter 2). Chapter 2 also includes background on microsensor networks, data aggregation algorithms, and cross-layer design. Chapters 3 and 4 discuss the LEACH protocol architecture for wireless microsensor networks. Chapter 3 describes the motivation for the design decisions, analysis of some features of LEACH, and comparison protocols that are similar to LEACH but use different set-up algorithms. Chapter 4 presents the simulation models and analytic and simulation results of our research. The simulations compare LEACH with several other protocols in terms of system lifetime, quality, and latency of data transfers. After describing our research on protocol architectures for wireless microsensor networks, we discuss in Chapter 5 an application-specific protocol architecture for delivery of MPEG-4 compressed video over wireless links. This chapter describes the design and evaluation of an unequal error protection scheme and experiments comparing unequal error protection with equal error protection. This dissertation ends with conclusions and a discussion of future directions in Chapter 6.

Chapter 2

Background

As illustrated by the explosive growth in cellular subscribers over the past couple of years, users want the flexibility and mobility that come with tetherless networks. At the same time, device technology has improved to the point where today's devices are smaller, cheaper, and more energy-efficient than in the past. Such portable devices with wireless networking capabilities allow us to get closer to the goal of "anytime, anywhere" connectivity to voice, video and data services.

The wireless channel presents several networking challenges not found in the wired domain, due to the limited resources of the channel and the portable device[1]. The challenges in designing network protocols, therefore, are to overcome these limitations:

- Limited channel bandwidth. The Federal Communications Commission (FCC) regulates what bandwidth particular networks can access and with how much power nodes are allowed to transmit. This limits the amount of bandwidth that can be given to each user of the network, requiring bandwidth-efficient protocols to maximize utility of the network.

- Limited node energy. Devices that access the wireless channel are often portable, obtaining energy from a local battery. This limits the amount of energy available to the node, affecting the lifetime. Protocols should therefore try to minimize energy dissipation to maximize node lifetime.

- Electromagnetic wave propagation. The radio wave is scattered as it propagates through the environment. Therefore, the power in the wave at the receiver (and hence the signal-to-

[1]While battery technology has improved to the point where standard nickel cadmium (NiCd) batteries can provide 40-50 Watt-hours/kg (144-180 J/g) [30] and advanced lithion ion (Li-Ion) batteries offer up to 120 Watt-hours/kg (432 J/g) [50], battery technology cannot keep up with the increasing demand for longer system lifetimes.

noise ratio (SNR)) depends on the distance between the transmitter and receiver (in addition to environmental factors). This precludes the use of collision detection to determine if two nodes are trying to access the channel at the same time. Thus, it is necessary to create clearly-defined media access control (MAC) protocols to minimize collisions.

- Error-prone channel. Errors can be caused by the environment, e.g., when buildings or cars block a direct, line-of-sight path between the transmitter and the receiver. In addition, collisions can occur between messages from different nodes.

- Time-varying conditions. The errors on the channel will vary over time, as the environment in which the transmitter and receiver are located changes and different nodes begin and end their own transmissions.

- Mobile nodes. Node mobility creates routing difficulties as nodes move in and out of communication range with each other.

There are general ideas that can be used to overcome these limitations. Low-energy protocols will help extend the limited node energy. Power control can be used to combat the radio wave attenuation—a transmitter can set the power of the radio wave such that it will be received with an acceptable power level. Link-layer protocols and MAC protocols can be used to combat channel errors. Finally, adaptive routing, MAC, and link-layer protocols can be used to overcome the time-varying conditions of the wireless channel and node mobility.

2.1 General-Purpose Layered Architectures

The past several years have shown a wealth of new protocols for wireless networks, including both routing and MAC protocols. Several standards have been proposed to facilitate interoperability among different devices in a wireless network. For example, the IEEE 802.11 [24] standard specifies a MAC protocol that was designed to minimize the probability of collision. Emerging standards such as HomeRF [43, 55] and Bluetooth [12, 36] specify the entire wireless network stack. Typically, the stack layers are implemented independently. This approach allows the communication architecture to be broken into smaller pieces, with each layer operating independently and providing some known support to the layers above. The main layers that will be presented here are the link-layer, MAC, and network (routing) layers.

Figure 2-1: Block diagram of a data transmission system.

2.1.1 Link-Layer Protocols

The limited bandwidth of the wireless channel combined with radio propagation loss and the broadcast nature of radio transmission make communication over a wireless channel inherently unreliable. Link-layer protocols are used to add information bits to the data bits to protect them against channel errors. Forward error correction (FEC) protocols add controlled redundancy to the data, in order to enable reliable transmission of data over unreliable channels. Typical channel coding systems contain a source coder, to reduce (with the goal of removing) redundancy from the data, followed by a channel coder, that adds controlled redundancy to the compressed data. The channel-coded data are sent over a channel, where noise is added to the stream. The channel decoder at the receiver produces an estimate of the source-coded stream, which is sent to the source decoder to extract the data to be given to the application. This system is depicted in Figure 2-1.

The two basic types of channel coders for FEC are block coders and convolutional coders. Block coders take a block of size k and produce a coded block of size n that depends only on the information in that block. This produces an (n, k) block code, where there are 2^k possible input blocks and 2^n possible output blocks. Convolutional coders also take blocks of size k as input and produce a coded block of size n. However, the output symbol is a function of not only the input block but of the last m input blocks. This represents an (n, k, m) convolutional code with memory order m [58]. For both block and convolutional codes, the code rate is $R = \frac{k}{n}$.

Convolutional encoding of data is performed by convolving k bits of the input with n generator polynomials to produce a rate-$\frac{k}{n}$ code, where $k \leq n$. Figure 2-2 shows a rate-$\frac{1}{2}$ coder. The added redundancy is used at the decoder to detect and correct a certain number of errors, using, for example, Viterbi decoding. Convolutional encoders are typically implemented using shift registers. If $k = 1$, the input stream can be continuously fed into the shift registers and the output can be continuously read at $\frac{n}{k}$ times the rate of the input.

Convolutional codes have the nice property that several different rate codes can be achieved

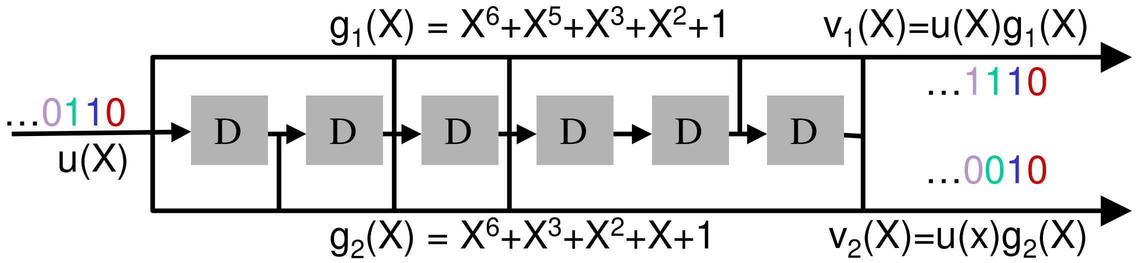
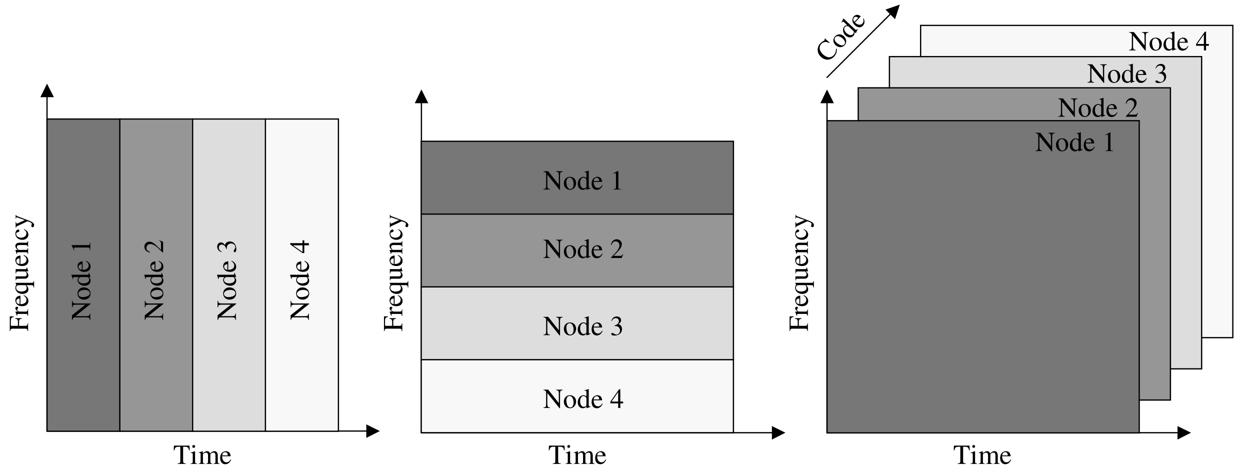


Figure 2-2: To achieve a rate-$\frac{1}{n}$ convolutional code, the source bits, $u(X)$, are convolved with generator functions, $g_1(X), ..., g_n(X)$. This figure shows a 6-memory (64 state) rate-$\frac{1}{2}$ encoder. An RCPC code is obtained by puncturing the output of a rate-$\frac{1}{n}$ code. For example, to get a rate-$\frac{2}{3}$ code from the above rate-$\frac{1}{2}$ convolutional coder, the output is punctured as follows: ...0110 $\rightarrow$ ...010$\cancel{1}$110$\cancel{0}$ $\rightarrow$ ...010110, where the crossed-out bits denote the punctured or discarded bits.

using the same *mother code*, so a single hardware implementation can produce varying amounts of protection. A rate-$\frac{a}{b}$ code can be achieved by puncturing, or discarding, the output bits from a rate-$\frac{1}{n}$ code. For every a input bits to the rate-$\frac{1}{n}$ coder, $(na-b)$ of these bits are discarded. The remaining b bits are sent as the channel coded signal. Rate compatible punctured convolutional (RCPC) [37] encoding is a special type of puncturing where higher rate codes are subsets of lower rate codes. Searches have been performed to determine the best generator and puncturing matrices [37].

2.1.2 Media Access Control (MAC) Protocols

MAC protocols are used to create predefined ways for multiple users to share the channel. There are 2 fundamentally different ways to share the wireless channel bandwidth among different nodes [69]: fixed-assignment channel-access methods (e.g., time-division multiple access (TDMA), frequency-division multiple access (FDMA), code-division multiple access (CDMA), and space-division multiple access (SDMA)) and random access methods (e.g., IEEE 802.11, carrier sense multiple access (CSMA), multiple access collision avoidance (MACA), and MACA for wireless (MACAW)[11]). Fixed-assignment MAC protocols allocate each user a given amount of bandwidth, either slicing the spectrum in time (TDMA), frequency (FDMA), code (CDMA), or space (SDMA). Since each node is allocated a unique part of the spectrum, there are no collisions among the data. However, fixed-assignment schemes are inefficient when all nodes do not have data to send, since scarce resources are allocated to nodes that are not using them. Random-access methods, on the other hand, do not assign users fixed resources. These are contention-based schemes, where nodes that have information to transmit must try to obtain bandwidth while minimizing collisions with other nodes' transmissions. These MAC protocols are more efficient than fixed-assignment MAC pro-



(a) TDMA (b) FDMA (c) CDMA

Figure 2-3: Fixed-assignment media access protocols. (a) Time-division multiple access (TDMA). In this protocol, each node is given the entire bandwidth for a certain time-slot. During this time-slot, no other node should transmit data. (b) Frequency-division multiple access (FDMA). In this protocol, each node is given a slice of bandwidth and continuously sends data within this bandwidth slice. No other node should transmit in the frequency slice given to a particular node. (c) Code-division multiple access (CDMA). In this protocol, each node spreads its data using a unique pseudo-random noise sequence. Therefore, all nodes use the entire bandwidth at all times.

tocols when nodes have bursty data. However, they suffer from possible collisions of the data, as all nodes are contending for the resources. Often protocols use a hybrid approach, e.g., combining TDMA and FDMA by allocating a certain time and frequency slot for each node. MAC protocols can be evaluated in terms of energy dissipation, fairness, and throughput, where the protocol is typically optimized to minimize energy dissipation, give each node its fair share of the bandwidth, and achieve high throughput [20, 82].

Time-Division Multiple Access (TDMA)

In TDMA, time is broken into *frames*, and each node is given a specific time-slot within the frame in which to transmit its data (see Figure 2-3a). During this time-slot, no other node should access the channel, so there are no collisions and the throughput is equal to the total data transmitted by each node. Assume there are N nodes that have data to transmit and the time for each frame is t_f and the channel bandwidth is B_w. Each node will get $t_s = \frac{t_f}{N}$ seconds in which to transmit data. Assuming a 1 bit/sec/Hz signaling scheme, each node can transmit $B_w t_s = B_w \frac{t_f}{N}$ bits per frame or $R_b = \frac{B_w}{N}$ bps.

Transmission occurs in bursts in TDMA, which reduces energy dissipation compared to a protocol where the transmission is continuous because the transmitter hardware (e.g., the phase-locked loops for frequency generation, the power amplifier) can be turned off when the node is not transmitting. In addition, this is a simple protocol to implement in the radio hardware. However, this protocol requires that some node have knowledge of all the transmitting nodes to create the schedule, and if the number of nodes that need to transmit data is variable, the schedule must be changed often, adding significant overhead to the protocol. In addition, using TDMA requires that all nodes be time-synchronized and requires guard slots to separate users [75]. These extra bits add to the overhead of a TDMA system.

Frequency-Division Multiple Access (FDMA)

In FDMA, the bandwidth is divided into slices such that each user gets a unique section of bandwidth in which to transmit data. Because no node is supposed to transmit in the bandwidth slice given to another node, there are no collisions between nodes' data. If there are N nodes that must share the B_w bandwidth, each node gets a frequency slice of size $\frac{B_w}{N}$ in which to transmit. As expected, given the same bandwidth B_w, the same number of users N, and the same signaling scheme (1 bit/sec/Hz), both FDMA and TDMA give each user the same bitrate ($R_b = \frac{B_w}{N}$ bps) under ideal conditions.

Transmission is continuous in FDMA, reducing the number of guard and synchronization bits needed compared to TDMA, thereby decreasing overhead (although FDMA may require guard bands to ensure transmissions do not overlap in frequency). However, FDMA requires that the transmitter hardware be on at all times, increasing the energy dissipation compared with a burst transmission protocol such as TDMA. In addition, FDMA requires good filtering to ensure that energy transmitted in the neighboring slices of bandwidth do not interfere with the transmission. FDMA also requires that some node have knowledge of all the transmitting nodes to allocate bandwidth appropriately.

In both TDMA and FDMA systems, the bandwidth is pre-allocated (separated in time or frequency for each user). The advantage of pre-allocation of the limited resources is that no collisions will occur, since each node has a unique time/frequency slice of bandwidth in which to transmit its data. The disadvantage of pre-allocation is that if nodes do not have data to send, the resources allocated to that node are wasted. To avoid waste, schedules can be changed often, reallocating time or frequency as needed. The problem with this is that it adds significant overhead to the protocols, as the controlling node must poll all the nodes to find out which ones have data to transmit, it must appropriately allocate resources, and it must transmit the schedule to all nodes.

Code-Division Multiple Access (CDMA)

Using direct-sequence spread spectrum (DS-SS) or CDMA, several nodes can transmit at the same time using the same bandwidth by spreading their data using a unique spreading code (typically a pseudo-random noise sequence). Reception of the signal is done by correlating with the spreading code used to transmit that signal. All other signals will appear as noise after de-correlation with the correct spreading code. Power control, where each node sets its transmit power to ensure the same amount of power at the receiving node, is very important in CDMA systems. If all nodes transmit at the same power, signals from nodes close to the receiving node will drown out signals from nodes further away from the receiving node. This is known as the *near-far* problem. In addition to reducing interference among different signals, using power control minimizes energy dissipation at the nodes.

As more nodes transmit data using their unique spreading code, the signal-to-noise ratio (SNR) of each transmission is reduced, whereas if fewer nodes transmit data, the SNR of each transmission is increased. Therefore, the performance degrades slowly as the number of nodes is increased [75]. The SNR will depend on the amount of spreading and the number of interfering signals. The number of simultaneous transmissions in a CDMA system is [32]:

$$N = \eta_b c_d \times \frac{B_w}{R_b(\frac{E_b}{I_o})} \tag{2.1}$$

where η_b is the bandwidth efficiency factor, c_d is the capacity degradation factor due to imperfect automatic gain control, B_w is the total bandwidth, R_b is the information rate, and $\frac{E_b}{I_o}$ is the bit energy to interference ratio required to achieve an acceptable probability of error. Assuming ideal conditions, this equation simplifies to:

$$N = \frac{B_w}{R_b \frac{E_b}{I_o}} \tag{2.2}$$

The minimum SNR required ($\frac{E_b}{I_o} = 1 = 0$ dB) is achieved when the minimum spreading of the data is performed. In this case, $R_b = \frac{B_w}{N}$. Therefore, under ideal conditions with minimum spreading, each of the N users can transmit at the same bitrate as in the TDMA and FDMA systems.

Space-division multiple access (SDMA)

New MAC protocols are exploiting the use of multiple transmit and receive antennas to increase system capacity using SDMA. Smart antennas are antenna arrays that use beamforming techniques to point the receiver towards a particular transmitter while nulling out other transmitters and the effects of multipath from all the transmitters [76]. Thus the receiver is no longer omni-directional but is capable of selecting the direction of the desired transmitter. Another SDMA protocol, the Bell Labs Adaptive Space-Time (BLAST) protocol, uses antenna arrays at both the transmitter and the receiver to point not only the receiver but also the transmitter [29]. The drawback of these systems is the high complexity and cost of having multiple transmit and/or receive antennas.

Random-Access Methods

In contrast with fixed-assignment multiple access approaches, in random-access methods, the resources are not assigned to individual nodes. When a node has data to send, it must contend for access to the channel with other nodes that have data to send. Therefore, collisions can occur when two nodes think that they have access to the channel at the same time. This, in turn, will reduce overall throughput. The goal with random-access approaches is to minimize collisions (and hence maximize throughput) while maintaining fairness in the use of resources among all the nodes.

Carrier-sense multiple access (CSMA) is one such random-access approach. Using CSMA, when a node has data to send, it listens to the channel to try to determine if any other node is currently transmitting. There are several different protocols that are followed if the channel is busy or idle [69]. In 1-persistent CSMA, the node keeps listening until the channel is free and then transmits its message. In non-persistent CSMA, after sensing a busy channel, the node enters a back-off state by setting a randomized timer. When the timer expires, the node again senses the channel. If it is busy, the node resets the timer. If the channel is free, the node transmits the packet. In p-persistent CSMA, the channel is assumed to be slotted. The node keeps sensing the channel until it is free, at which point the node transmits its packet during the first available free slot with probability p. With probability $1-p$, the node waits until the next slot and again senses the channel. If the channel is free, the node transmits its packet during the first available free slot with probability p and waits until the next slot with probability $1-p$. This continues until the node sends its packet or senses a busy channel, at which point the node must wait until the channel is free and begin the process again.

Using carrier-sense will reduce collisions, but it cannot guarantee that collisions will not occur. For example, there is a small (but nonzero) probability that two nodes will sense the channel at the same time, both decide that the channel is free for transmission, and both transmit data at the same time, causing a collision of both data messages. The more likely collisions will occur because the transmitting node cannot "hear" everything that the receiving node can hear. In this case, the transmitting node assumes that the channel is free for transmission, while the receiving node is busy receiving data from another node. This will cause a collision at the receiving node, which will not be able to receive data from either transmitting node. This is known as the *hidden terminal* problem [84]. The main problem is that collisions occur at the receiver but need to be detected at the transmitter. In a wired network, the transmitter and receiver hear the same message and the transmitter can detect a collision at the receiver. In a wireless network, on the other hand, due to the propagation loss of the radio signal, the transmitter cannot always hear the same message the receiver hears.

Newer randomized protocols, such as MACAW and the IEEE 802.11 standard, use pre-transmission messages (e.g., request-to-send (RTS) and clear-to-send (CTS)) to further reduce the risk of collision. The transmitting node sends out an RTS message, and the receiving node must send out a CTS message before the transmitting node can send any data. All nodes listen for CTS messages. Once a CTS message is heard, all other nodes know not to transmit their own RTS messages on the channel until the current transmission is complete. This further reduces collisions, at the expense of increased overhead and energy dissipation since nodes need to receive all messages to obtain the CTS messages.

In addition to collisions, there may be packet losses if the transmitter has to back off too long and the data are time-sensitive. There can also be buffer overflow if the node has too many packets to send and cannot access the channel, resulting in lost data.

Randomized protocols have the advantage of being simple to implement, requiring no knowledge of the network topology or global control, and they are easy to configure. However, there are several disadvantages to these protocols. First, in large shared-media networks with high utilization, collisions will occur. Second, this protocol is not optimized for energy efficiency. Since the radio electronics dissipate energy, carrier-sense and receive operations are expensive. With no coordination among the nodes, random access MAC protocols require that the radio electronics be on more often than is necessary to transmit and/or receive a message.

Table 2.1 shows the media access used in several wireless systems. Systems that must guarantee a certain quality of service (QoS) to the user, such as cellular systems, typically use fixed-assignment multiple access techniques. In contrast, systems that make no guarantees about timely delivery of data, such as wireless data networks, often use random-access techniques. Several systems combine media access technologies, such as using TDMA with a CDMA protocol (e.g., PCS-2000 [32]),

Table 2.1: Media access used in different wireless systems.

System	Media access
AMPS	FDMA
IS-95	DS-CDMA/FDMA
PACS	TDMA
PCS-2000	CDMA/TDMA
GSM	TDMA
Lucent WaveLAN	802.11
Bluetooth	FH-SS/TDMA
HomeRF	FH-SS/TDMA/CSMA

assigning slots to users within a given frequency-hoping spread-spectrum (FH-SS) protocol (e.g., Bluetooth [12, 36]), or using either TDMA or CSMA within a FH-SS protocol (e.g., HomeRF, where TDMA is used for real-time data delivery and CSMA is used for asynchronous delivery) [43, 55].

2.1.3 Routing Protocols

Routing protocols for wireless networks can broadly be classified into two categories: multi-hop routing protocols and cellular/clustering approaches.

Multi-Hop Routing

Routing protocols for wired networks fall into two categories: distance vector routing and link-state [45]. Distance vector approaches route packets by having each node advertise distances to their neighbors, who then choose the shortest path to a given destination and store this information in a routing table. As a packet comes to the node, it looks in its routing table to determine the next hop to get the packet to its destination. Link-state protocols, on the other hand, have nodes disseminate the entire topology map and have the individual nodes use a shortest path algorithm (such as Dijkstra's Algorithm) to find the best route to a given destination. These routing approaches have been incorporated into wireless networks using minor modifications, resulting in destination-sequenced distance vector (DSDV) and ad hoc on-demand distance vector (AODV) routing protocols [72, 71]. However, there are problems with using these routing approaches in wireless networks. The periodic messages needed to maintain valid routes may not only congest the network, they may also drain the limited battery supply of a portable node. Dynamic source routing (DSR), solves this problem by only creating routes on an on-demand basis [14]. This minimizes the amount of overhead needed in creating routes, at the expense of latency in finding a route when it is needed. These are *ad-hoc*, self-configuring protocols that are robust to node failures.

Work has been done on "minimum-energy" routing protocols to try to extend the lifetime of the portable devices in a wireless network. For example, [62] discusses a strategy for choosing multi-hop routes to minimize the power dissipated in the nodes along the route. In these approaches, an intermediate node is used as a hop if and only if it minimizes the total energy compared with not using this middle hop node. A similar idea is proposed in [82]. In this work, the authors note that in a wireless network, transmission between neighboring nodes causes interference, which can degrade performance. Hence, they choose routes to minimize energy dissipation subject to a minimum interference criterion.

Another routing protocol, the Self-Organizing Wireless Adaptive Network (SWAN) protocol [81], uses dynamic topology management with power control to "deform [the network] gradually instead of [having the network] periodically broken down and rebuilt." This allows user data to experience a minimum amount of delay and no outages due to network recovery functions.

Recently, there has been much work on "power-aware" routing protocols for wireless networks [59, 70, 85]. In these protocols, optimal routes are chosen based on the energy at each node along the route. Routes that are longer but use nodes with more energy than the nodes along the shorter routes are favored. This helps avoid "hot-spots" in the network, where a node is often used to route other nodes' data, and it helps to evenly distribute energy dissipation.

One method of choosing routes is to use "minimum transmission energy" (MTE) routing [28, 84]. In this approach, the intermediate nodes are chosen such that the sum of squared distances (and hence the total transmit energy, assuming a d^2 power loss) is minimized; thus for the configuration shown in Figure 2-4, node A would transmit to node C through node B if and only if:

$$E_{transmit}(d = d_{AB}) + E_{transmit}(d = d_{BC}) < E_{transmit}(d = d_{AC}) \tag{2.3}$$

or, assuming a $\frac{1}{d^2}$ attenuation model,

$$d_{AB}^2 + d_{BC}^2 < d_{AC}^2 \tag{2.4}$$

We implemented a multi-hop routing approach to compare with LEACH. When using a routing approach with sensor networks, all the nodes must get their data to the base station. Each node

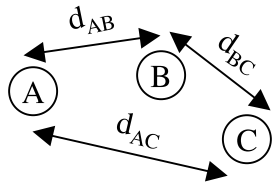


Figure 2-4: In minimum transmission energy (MTE) routing, node A would transmit to node C through node B if $E_{transmit}(d = d_{AB}) + E_{transmit}(d = d_{BC}) < E_{transmit}(d = d_{AC})$ or $d_{AB}^2 + d_{BC}^2 < d_{AC}^2$.

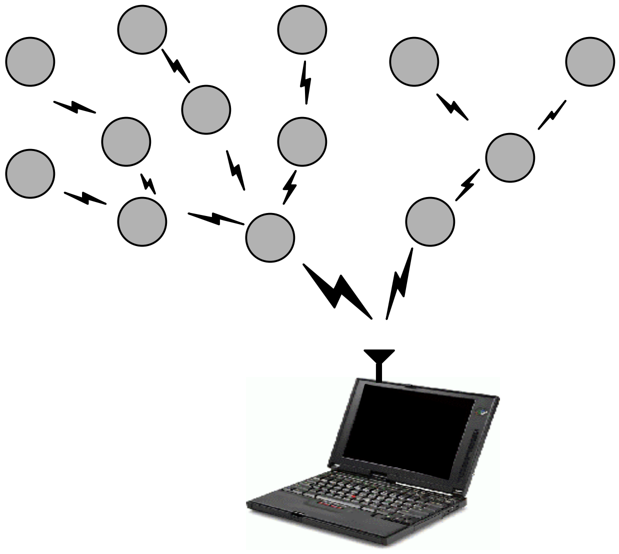


Figure 2-5: In MTE routing applied to a microsensor network, data are passed along the routes until they reach the base station.

runs a start-up routine to determine its next-hop neighbor, as determined by the particular routing protocol. For our simulations, we implemented an MTE approach to choosing routes, as this minimizes the transmit energy required to get the data to the base station. Data are passed to each node's next-hop neighbor until the data reaches the base station, as shown in Figure 2-5. As nodes run out of energy, the routes need to be recomputed to ensure connectivity with the base station.

Clustering

Another method of wireless communication is to use a clustering approach, similar to a cellular telephone network. In this case, nodes send their data to a central *cluster-head* that forwards the data to get it closer to the desired recipient. Clustering enables bandwidth reuse and can thus increase system capacity. Using a clustering approach enables better resource allocation and helps improve power control [54]. In addition, the hierarchical structure obtained using clustering can help overcome some of the problems with node mobility.

While conventional cellular networks rely on a fixed infrastructure [32, 69], new research is focusing on ways to deploy clustering architectures in ad-hoc fashion, without the assistance of a fixed infrastructure [7, 54, 57, 60, 96]. Early work by Baker *et al.* developed a linked cluster architecture [7]. Using the distributed linked cluster algorithm (LCA), nodes are assigned to be either ordinary nodes, cluster-head nodes, or gateways between different clusters. The cluster-head acts as a local control center, whereas the gateways act as the backbone network, transporting data between clusters. This enables robust networking with point-to-point connectivity.

A similar system, the Near Term Digital Radio (NTDR) [78, 96] uses a clustering approach with a two-tier hierarchical routing algorithm. Nodes form local clusters, and intra-cluster data are sent directly from one node to the next, whereas inter-cluster data are routed through the cluster-head nodes. This design allows for increased system capacity by reducing interference. In NTDR networks, the cluster-head nodes change as nodes move in order to keep the network fully connected. This protocol, designed to be used for a wireless data network, enables point-to-point connectivity.

Lin *et al.* develop a fully distributed cluster formation and communication algorithm where there are no fixed cluster-head nodes in the cluster [57]. This has the advantage of avoiding "hot-spots", or bottlenecks in the network. Their distributed cluster formation uses a lowest-node-ID algorithm, whereby the cluster-head position is assigned to the node with the lowest of its ID and all its neighbors IDs. A cluster maintenance algorithm is created to ensure connectivity of all nodes in the presence of node mobility, and a combination TDMA/CDMA scheme is used to ensure minimum inter- and intra-cluster interference.

Power control can be used to dynamically adjust the size of clusters [54]. If open-loop power control is used, the cluster-head node sends out a beacon, and nodes that hear the beacon join the cluster. If there are too many nodes in the cluster, the cluster-head can reduce the beacon signal strength so fewer nodes will hear it. On the other hand, if the cluster is too small, the cluster-head can increase its beacon signal strength to increase cluster membership. New clusters may be formed when a cluster-head decreases its membership size, and clusters may be merged when a cluster-head increases its membership size in order to keep the network fully connected.

McDonald *et al.* develop a clustering algorithm that enables good routing (high probability of path availability) while supporting node mobility and stability [60]. Their (α,t) cluster algorithm creates clusters of nodes where the probability of path availability is bounded over time. This allows the clustering algorithm to adapt to node mobility, creating more optimal routing under low node mobility.

In a static clustering protocol for microsensor networks, nodes are organized into clusters initially, and these clusters and the cluster-heads remain fixed throughout the lifetime of the network (see Figure 2-6). Nodes transmit their data to the cluster-head node during each frame of data transfer, and the cluster-head forwards the data to the base station. Since data from nodes located close to each other are highly correlated, the cluster-head node aggregates the signals to reduce the actual amount of data that must be transmitted to the base station. Since the cluster-head must transmit the data to the end-user via the shared wireless channel, if the cluster-head could not aggregate the data, there would be no advantage to using this approach over an approach where each node sent its data directly to the base station.

2.2 Microsensor Networks

Since both device and battery technology have only recently matured to the point where microsensor nodes are feasible, this is a fairly new field of study. Researchers have begun discussing not only the uses and challenges facing sensor networks [5, 27, 73], but also have been developing preliminary ideas as to how these networks should function [19, 22, 40, 48] as well as the appropriate low-energy architecture for the sensor nodes themselves [17, 25, 74].

Sensor nodes typically contain a sensor module, some sort of processing element, and a wireless interface module [19, 25]. As these nodes are battery-operated, it is important to ensure each of these modules is low-power to extend node lifetime. This can be accomplished using techniques such as dynamic voltage scaling (DVS) [64] and operating system power-management [86]. Using Low Power Wireless Integrated Microsensor (LWIM) technology, Dong *et al.* showed that by exploiting the unique characteristics of sensor data processing requirements, a low-power signal processing

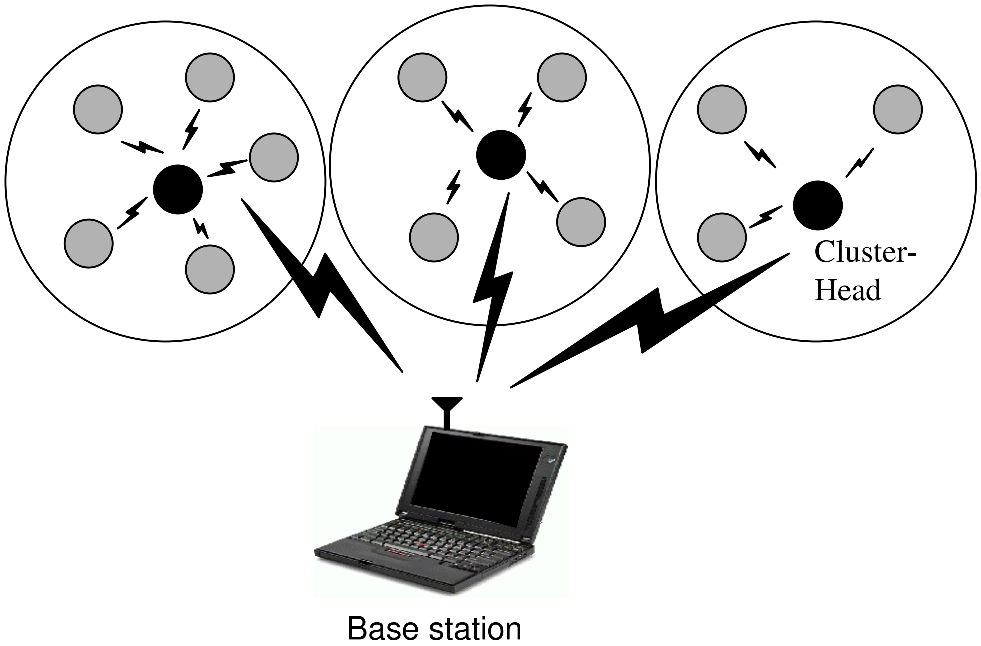


Figure 2-6: In static clustering in a microsensor network, data are passed from the nodes to the cluster-heads, who forward the data to the base station.

subsystem for sensor nodes can be designed to operate at less than 3 μW [25].

In addition to developing low-energy hardware, it is important that microsensor networks use low-energy protocols. With this goal in mind, the authors in [41] developed SPIN (Sensor Protocols for Information via Negotiation), a family of protocols to disseminate information in a wireless microsensor network. In SPIN, large data messages are named using high-level data descriptors, called meta-data. Nodes use meta-data negotiation to eliminate the transmission of redundant data throughout the network. Allowing nodes to base routing decisions on application-specific information about the data enables large energy-savings compared with conventional approaches.

Another low-energy protocol architecture was developed by Clare *et al.* [22] as part of the AWAIRS (Adaptive Wireless Arrays for Interactive Reconnaissance, Surveillance, and Target Acquisition in Small Unit Operations) microsensor system [6]. This protocol enables self-organizing sensor networks and uses a TDMA MAC approach for low-energy operation. This protocol forms the network by having each node join the existing network, where the first two nodes alive form the initial network. Each node is given several TDMA slots in which to transmit point-to-point to each of its neighbors as well as broadcast to all of its neighbors, and each node also knows when it must be awake to receive data (either sent unicast or broadcast) from all of its neighbors. The authors claim that this allows the nodes to remain in the sleep state, with radios powered-down, for a large amount of time.

Recent work by Intanagonwiwat *et al.* [48] describes a completely different paradigm for microsensor communication: *directed diffusion*. Using directed diffusion, data are named with attribute-value pairs, and interests for certain types of data are disseminated throughout the network. These interests diffuse to the correct area, setting up gradients that draw events of interest back to the node that originated the request. Good routes are inherently reinforced, enabling low-energy routing of the data. In addition, data aggregation and caching can be performed within the network, further reducing node energy dissipation.

Several institutions have begun large-scale projects to develop system and protocol architectures for wireless microsensor networks. These projects include:

- AWAIRS: Adaptive Wireless Arrays for Interactive Reconnaissance, Surveillance, and Target Acquisition in Small Unit Operations (UCLA/Rockwell Science Center) [6, 22]
- WINS: Wireless Integrated Network Sensors (UCLA/Rockwell Science Center) [74, 97]
- Smart Dust: Autonomous Sensing and Communication in a Cubic Millimeter (U.C. Berkeley) [51, 87]
- SCADDS: Scalable Coordination Architectures for Deeply Distributed Systems (USC/ISI) [27, 48, 79]
- μ-AMPS: Micro-Adaptive Multi-domain Power-aware Sensors (MIT) [40, 42, 65]

In addition, there are numerous projects to develop "ubiquitous computing" architectures. Researchers predict that the future of computing is one where computers are everywhere but at the same time invisible to the user [13, 95]. Distributed and embedded microsensor networks (as well as actuator networks) will be essential technology to enable the full integration of computers into our everyday lives.

2.2.1 The MIT μ-AMPS Project

Researchers at MIT have started the μ-AMPS (Micro-Adaptive Multi-domain Power-aware Sensors) project to develop a framework for implementing adaptive energy-aware distributed microsensors [65]. The protocol architectures for microsensor networks described in this dissertation were designed in the context of the μ-AMPS project. In this project, we assume that the basic sensor technology is available and focus instead on the signal and power conditioning, communication, and collaboration aspects of system design. The goal of the μ-AMPS project is to provide an energy-efficient and scalable solution for a range of sensor applications. This involves designing innovative energy-optimized solutions at all levels of the system hierarchy including: physical layer (e.g., transceiver design), data-link layer (packetization and encapsulation), media access layer (multi-user communication with emphasis on scalability), network/transport layer (routing and aggregation schemes), session/presentation layer (real-time distributed OS), and application layer (innovative applications).

The architecture for a μ-AMPS node is shown in Figure 2-7. The first version of the μ-AMPS node contains off-the-self components, including the StrongARM (SA-1100) microprocessor running a lean version of the RedHat eCos operating system for implementation of DSP algorithms and communication protocols [31]. This board serves as a proof-of-concept for the networked microsensor system. The μ-AMPS nodes sense data (using either a microphone or geophone sensor), filter and digitize the data, perform signal processing functions on the data, and transmit the data. On the receiving end, the nodes receive data, perform signal processing functions and transmit a response. Sensor network protocols can be implemented within this μ-AMPS framework. The use of the SA-1100 processor allows the μ-AMPS nodes to be easily programmed to run different protocols and enables monitoring of the energy dissipation required for the various functions performed within the protocol.

The next generation μ-AMPS nodes will be custom-designed, including application-specific integrated circuits (ASICs) designed by different members of the μ-AMPS project team. These changes include replacing the StrongARM SA-1100 with a custom DSP, adding data aggregation algorithm ASICs, replacing the off-the-shelf radio with a custom-made radio transceiver, and possibly adding a custom real-time controller chip for running the protocols. These final μ-AMPS nodes will display the energy and size efficiency needed for microsensor networks, and the μ-AMPS framework will allow easy testing and analysis of different protocols, algorithms, and applications for wireless microsensor networks.

2.2.2 Sensor Data Aggregation

Sensor data are different from the data associated with traditional wireless networks in that it is not the actual data itself that is important; rather, the analysis of the data, which allows an end-user to determine something about the environment that is being monitored, is the important result of

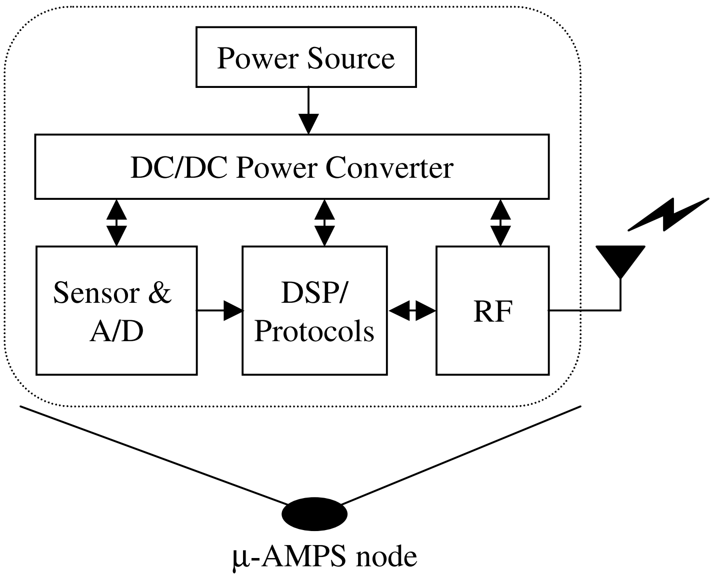


Figure 2-7: Architecture of a μ-AMPS node. Current μ-AMPS nodes consist of off-the-shelf components, including the StrongARM SA-1100 processor. Future revisions of this node will include custom-designed ASICs.

a sensor network. For example, if the sensors are monitoring an area for surveillance purposes, the end-user does not need to see the data from all the individual sensors but does need to know whether or not there has been an intrusion in the area being monitored. Therefore, automated methods of combining or *aggregating* the data into a small set of meaningful information are required [15, 38, 52, 93]. In addition to helping avoid information overload, data aggregation, also known as *data fusion*, can combine several unreliable data measurements to produce a more accurate signal by enhancing the common signal and reducing the uncorrelated noise. The classification performed on the aggregated data might be performed by a human operator or automatically. Both the method of performing data aggregation and the classification algorithm are application-specific.

In a conventional sensor network, all the data X are transmitted to the base station, where they are processed (aggregated) to receive the data $f(X)$. Automated methods can then used to classify this aggregate signal (e.g., based on template source signatures). However, the function f can sometimes be broken up into several smaller functions $f_1, f_2, ..., f_n$ that operate on subsets of the data $X_1, X_2, ..., X_n$ such that:

$$f(X) \approx g(f_1(X_1), f_2(X_2), ..., f_n(X_n)) \tag{2.5}$$

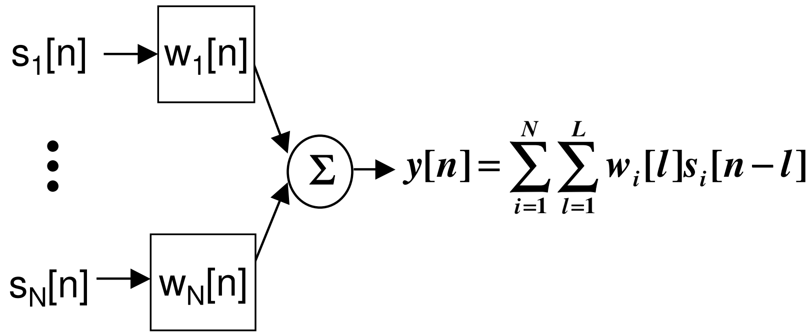


Figure 2-8: Block diagram of the beamforming algorithm. The individual sensor signals, $s_i[n]$ are filtered with weighting filters, $w_i[n]$ to get the beamformed signal, $y[n]$.

The ability to break the function in this manner leads to the possibility of the sensor nodes performing local computation on a subset of the data. This has the potential to greatly reduce the amount of information that needs to be transmitted to an end-user. If the energy required to perform the signal processing functions is less than the energy required to transmit the initial data, energy savings is achieved.

One method of aggregating data is called *beamforming* [68, 101]. Beamforming combines signals from multiple sensors as follows:

$$y[n] = \sum_{i=1}^{N} \sum_{l=1}^{L} w_i[l] s_i[n-l] \tag{2.6}$$

where $s_i[n]$ is the signal from the i^{th} sensor, $w_i[n]$ is the weighting filter for the i^{th} signal, N is the total number of sensors whose signals are being beamformed, and L is the number of taps in the filter. This algorithm is depicted in Figure 2-8. The weighting filters are chosen to satisfy an optimization criteria, such as minimizing mean squared error (MSE) or maximizing signal-to-noise ratio (SNR). Various algorithms, such as least mean squared (LMS) error approach and the maximum power beamforming algorithm [101], have been developed to determine good weighting filters. These algorithms have various energy and quality tradeoffs [94]. For example, the maximum power beamforming algorithm is capable of performing blind beamforming, requiring no information about the sensor locations. However, this algorithm is compute-intensive, which will quickly drain the limited energy of the node.

By determining the amount of computation needed to fuse the data from several sensor nodes and the associated energy and time costs to perform these signal processing operations, it is possible to determine the optimum tradeoff between computation and communication. For example, the authors in [21] determine the optimal number of sensors to utilize in processing data to minimize the overall time until the solution is achieved given a simple line network. In general networks, there is no closed-form solution as there is for a linear network, but heuristics can be developed to achieve good tradeoffs between computation and communication.

2.3 Cross-Layer Design

Designing protocols according to a layering approach enables the protocol designer to separately and independently design the different functions [103]. However, such an approach does not allow different layers to interact and therefore may not be optimal in all situations. Researchers have argued the need to collapse the stack and design completely integrated protocol architectures [2, 23]. Alternatively, modularity can be preserved while improving network performance by using a cross-layer design, where information is shared between layers. For example, exposing the application requirements to the lower layers of the protocol stack can greatly enhance the application-perceived quality of the network.

The Snoop protocol uses a cross-layer design to expose data loss information throughout the different layers [8]. This improves performance of mobile node connections by enabling a transport-aware link protocol and a link-aware transport protocol to help nodes determine whether losses occurred due to network congestion or wireless channel errors. A similar approach is followed by Inouye *et al.* for the design of a protocol architecture that supports mobile multimedia applications [46]. In this work, information in the form of *quasi-invariants*, *guards*, and *intelligent adaptation* is shared among the different stack layers in order to expose mobility problems to all the layers.

Recently, there has been a great deal of work on application-controlled routing [3, 41, 48]. Adjie-Winoto *et al.* developed an intentional naming system that allows users to specify application-specific names and uses an overlay network to perform name resolution and message routing [3]. Similarly, the SPIN [41] and directed diffusion [48] protocols use application-specific data naming and routing to achieve energy efficiency in a wireless microsensor network.

Finally, work with active networks has shown that networks protocols can be defined on an application-specific basis, where protocols are created by the applications to support the functions they require [53, 91]. These protocols are transmitted with the data via *Smart Packets* so that intermediate nodes can reconfigure themselves to support the appropriate protocol.

The protocol architectures described in this dissertation employ the technique of cross-layer design by exposing lower layers of the protocol stack to the requirements of the application. The research described in this dissertation illustrates the high performance that can be achieved despite the harsh conditions of the wireless channel using application-specific protocol architectures.

Chapter 3

The LEACH Protocol Architecture

Wireless microsensor networks will enable reliable monitoring of remote areas. These networks are essentially data-gathering networks where the data are highly correlated and the end-user requires a high-level description of the environment the nodes are sensing. In addition, these networks require ease of deployment, long system lifetime, and low-latency data transfers. This is a very different paradigm than traditional wireless networks that require point-to-point connectivity, have uncorrelated data, and often can rely on a fixed infrastructure. The limited battery capacity of microsensor nodes and the large amount of data that each node may produce translates to the need for high application-perceived performance at a minimum cost, in terms of energy and latency. A cross-layer or application-specific protocol architecture can meet these specifications by exposing lower layers of the protocol stack to the requirements of the application.

To meet the requirements of wireless microsensor networks, we developed LEACH (Low-Energy Adaptive Clustering Hierarchy), an application-specific protocol architecture (see Figure 3-1) [40]. LEACH is a clustering-based protocol that includes the following features:

- randomized, adaptive, self-configuring cluster formation,
- localized control for data transfers,
- low-energy media access, and
- application-specific data processing, such as data aggregation.

The application that typical microsensor networks support is the remote monitoring of an environment. Since individual nodes' data are correlated in a microsensor network, the end-user does not require all the (redundant) data; rather, the end-user needs a high-level function of the data

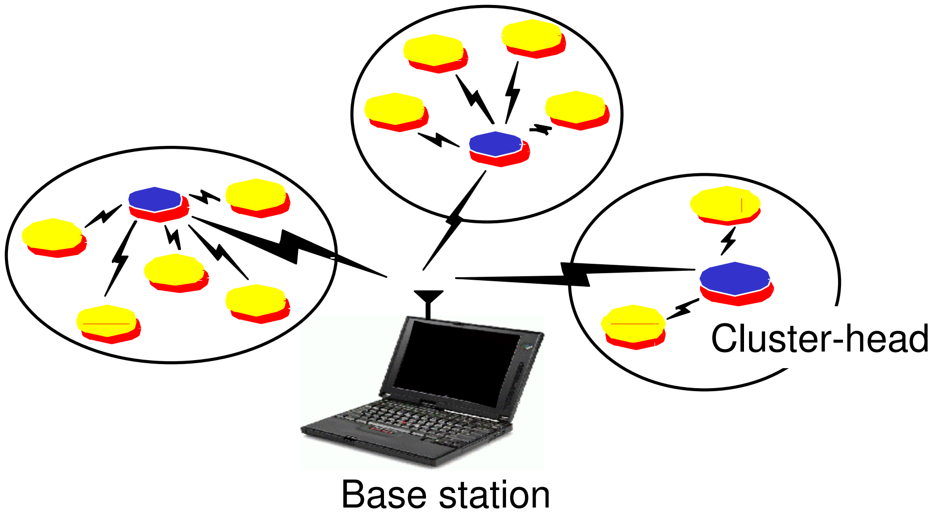


Figure 3-1: The LEACH protocol for microsensor networks. LEACH includes adaptive, self-configuring cluster formation, localized control for data transfers, low-energy media access, and application-specific data processing.

that describes the events occurring in the environment. Because the correlation is strongest among data signals from nodes located close to each other, we chose to use a clustering infrastructure as the basis for LEACH. This allows all data from nodes within the cluster to be processed locally, reducing the data set that needs to be transmitted to the end-user. In particular, data aggregation techniques can be used to combine several correlated data signals into a smaller set of information that maintains the *effective* data (i.e., the information content) of the original signals. Therefore, much less actual data needs to be transmitted from the cluster to the base station. If the cost in terms of energy dissipation of communicating data is greater than the cost of computing on the data, considerable energy savings can be achieved by locally aggregating a large amount of data into a smaller set of data before transmission to the base station.

In LEACH, the nodes organize themselves into local clusters, with one node acting as the *cluster-head*. All non-cluster-head nodes must transmit their data to the cluster-head, while the cluster-head node must receive data from all the cluster members, perform signal processing functions on the data (e.g., data aggregation), and transmit data to the remote base station. Therefore, being a cluster-head node is much more energy-intensive than being a non-cluster-head node. In the scenario where all nodes are energy-limited, if the cluster-heads were chosen a priori and fixed throughout the system lifetime, as in a static clustering algorithm, the cluster-head sensor nodes would quickly use up their limited energy. Once the cluster-head runs out of energy, it is no longer operational.

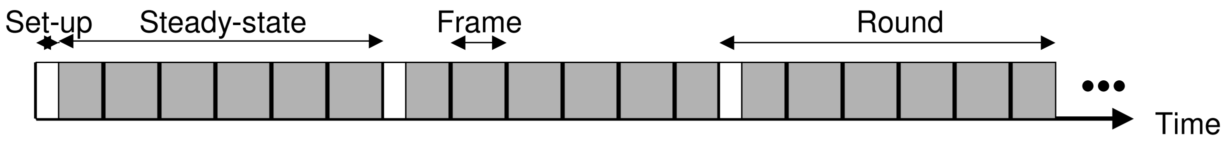


Figure 3-2: Time-line showing LEACH operation. Adaptive clusters are formed during the set-up phase and data transfers occur during the steady-state phase.

Thus, when a cluster-head node dies (e.g., uses up all its battery energy), all the nodes that belong to the cluster lose communication ability. Thus LEACH incorporates randomized rotation of the high-energy cluster-head position such that it rotates among the sensors in order to avoid draining the battery of any one sensor in the network. In this way, the energy load associated with being a cluster-head is evenly distributed among the nodes.

Media access in LEACH was chosen to reduce energy dissipation in the non-cluster-head nodes. Since the cluster-head node knows all the cluster members, it can create a TDMA schedule that tells each node exactly when to transmit its data. This allows the nodes to remain in the sleep state, with internal modules powered-down, as long as possible. In addition, using a TDMA schedule for data transfer prevents intra-cluster collisions.

The operation of LEACH is divided into *rounds*. Each round begins with a set-up phase when the clusters are organized, followed by a steady-state phase where several frames of data are transfered from the nodes to the cluster-head and on to the base station, as shown in Figure 3-2. The nodes must all be time-synchronized in order to start the set-up phase at the same time. In order to minimize the set-up overhead, the steady-state phase is long compared to the set-up phase.

3.1 Self-Configuring Cluster Formation

LEACH forms clusters by using a distributed algorithm, where nodes make autonomous decisions without any centralized control. The advantages of this approach are that no long-distance communication with the base station is required and distributed cluster formation can be done without knowing the exact location of any of the nodes in the network. In addition, no global communication is needed to set up the clusters, and nothing is assumed about the current state of any other node during cluster formation. The goal is to achieve the global result of forming good clusters out of the nodes, purely via local decisions made autonomously by each node.

3.1.1 Determining Cluster-Head Nodes

Given that we want to produce clusters using a distributed protocol, what are the important goals that we are trying to achieve? What constitutes a good cluster formation? To begin with, we want to design the algorithm such that there are a certain number of clusters, k, during each round. Second, we want to try to evenly distribute the energy dissipation among all the nodes in the network so that there are no overly-utilized nodes that will run out of energy before the others. This will maximize the time until the first node death. As being a cluster-head node is much more energy-intensive than being a non-cluster-head node (since the cluster-head node must receive data from all the nodes in the cluster, perform signal processing functions on the data, and transmit the data to an end-user who may be far away), evenly distributing the energy load among all the nodes in the network requires that each node take its turn as cluster-head. Therefore, the cluster formation algorithm should be designed such that nodes are cluster-heads approximately the same amount of time, assuming all the nodes start with the same amount of energy. Finally, we would like the cluster-head nodes to be spread throughout the network, as this will minimize the distance the non-cluster-head nodes need to send their data.

In LEACH, sensors elect themselves to be cluster-heads at the beginning of round $r+1$ (which starts at time t) with a certain probability, $P_i(t)$. This probability is chosen such that the expected number of cluster-head nodes for this round is k. Thus:

$$E[\#\ \mathrm{CH}] = \sum_{i=1}^{N} P_i(t) * 1 = k \tag{3.1}$$

where N is the total number of nodes in the network. Ensuring that all nodes are cluster-heads the same number of times requires each node to be a cluster-head once in $\frac{N}{k}$ rounds. Combining these constraints gives the following probability for each node i to be a cluster-head at time t:

$$P_i(t) = \begin{cases} \frac{k}{N - k*(r \bmod \frac{N}{k})} & : \quad C_i(t) = 1 \\ 0 & : \quad C_i(t) = 0 \end{cases} \tag{3.2}$$

where r is the number of rounds that have passed and $C_i(t) = 0$ if node i has already been a cluster-head in the most recent $(r \bmod \frac{N}{k})$ rounds and 1 otherwise. Therefore, only nodes that have not already been cluster-heads recently, and which presumably have more energy available than nodes that have recently performed this energy-intensive function, may become cluster-heads at round $r+1$. The expected number of nodes that have not been cluster-heads in the first r rounds is $N - k * r$. After $\frac{N}{k}$ rounds, all nodes are expected to have been cluster-head once, following which they are all eligible to perform this task in the next sequence of rounds. Since $C_i(t)$ is 1 if node i is eligible to be a cluster-head at time t and 0 otherwise, the term $\sum_{i=1}^{N} C_i(t)$ represents the total number of nodes that are eligible to be a cluster-head at time t, and

$$E[\sum_{i=1}^{N} C_i(t)] = N - k * (r \bmod \frac{N}{k}) \tag{3.3}$$

This ensures that the energy at each node is approximately equal after every $\frac{N}{k}$ rounds. Using Equations 3.2 and 3.3, the expected number of cluster-heads per round is:

$$\begin{aligned} E[\#\ \mathrm{CH}] &= \sum_{i=1}^{N} P_i(t) * 1 \\ &= (N - k * (r \bmod \frac{N}{k})) * \frac{k}{N - k * (r \bmod \frac{N}{k})} \\ &= k \end{aligned} \tag{3.4}$$

The optimal k can be determined analytically based on the energy dissipation models for computation and communication and the network topology. Analysis and simulation to determine the optimal k will be described in detail in Section 4.3.

This choice of probability for becoming a cluster-head is based on the assumption that all nodes start with an equal amount of energy, and that all nodes have data to send during each frame. If nodes begin with different amounts of energy (or an event-driven model is used, whereby nodes only send data when some event occurs in the environment), the nodes with more energy should be cluster-heads more often than the nodes with less energy, in order to ensure that all nodes die at approximately the same time. This can be achieved by setting the probability of becoming a cluster-head as a function of a node's energy level relative to the aggregate energy remaining in the network, rather than purely as a function of the number of times the node has been cluster-head:

$$P_i(t) = \frac{E_i(t)}{E_{total}(t)} k \tag{3.5}$$

where $E_i(t)$ is the current energy of node i, and

$$E_{total}(t) = \sum_{i=1}^{N} E_i(t) \tag{3.6}$$

Using these probabilities, the nodes with higher energy are more likely to become cluster-heads than nodes with less energy. The expected number of cluster-head nodes is:

$$\begin{aligned} E[\#\ \text{CH}] &= \sum_{i=1}^{N} P_i(t) * 1 \\ &= (\frac{E_1(t)}{E_{total}} + \ldots + \frac{E_N(t)}{E_{total}})k \\ &= k \end{aligned} \tag{3.7}$$

Equation 3.2 can be approximated by Equation 3.5 when the nodes begin with equal energy, E_o. If a node has been a cluster-head in the last $r < \frac{N}{k}$ rounds, its energy is approximately $E_o - E_{CH}$, where E_{CH} is a large number less than E_o. If the node has not been a cluster-head in the last r rounds, its energy is approximately E_o, since being a non-cluster-head node does not require much energy from the node relative to being a cluster-head node. Since it is expected that kr nodes have been cluster-heads and $N - kr$ nodes have not been cluster-heads in the last r rounds, the total expected energy is given by:

$$E[E_{total}] = E_o(N - kr) + (E_o - E_{CH})(kr) \tag{3.8}$$

Therefore, Equation 3.5 becomes:

$$E[P_i(t)] = \begin{cases} \frac{E_o k}{E_o(N-kr)+(E_o-E_{CH})kr} & : \quad C_i(t) = 1 \\ \frac{(E_o-E_{CH})k}{E_o(N-kr)+(E_o-E_{CH})kr} & : \quad C_i(t) = 0 \end{cases} \tag{3.9}$$

Since $E_o >> (E_o - E_{CH})$, this can be simplified to:

$$E[P_i(t)] \approx \begin{cases} \frac{k}{N-kr} & : \quad C_i(t) = 1 \\ 0 & : \quad C_i(t) = 0 \end{cases} \tag{3.10}$$

Thus the expected probability for each node becoming a cluster-head at round t is exactly the same as in Equation 3.2 (for $r < \frac{N}{k}$).

Using the probabilities in Equation 3.5 requires that each node have an estimate of the total energy of all the nodes in the network. This requires a routing protocol that allows each node to determine the total energy, whereas the probabilities in Equation 3.2 enable each node to make completely autonomous decisions. One approach to avoid the routing protocol might be to approximate the aggregate network energy by averaging the energy of the nodes in each cluster and multiplying by N.

3.1.2 Set-up Phase

Once the nodes have elected themselves to be cluster-heads using the probabilities in Equation 3.2 or 3.5, the cluster-head nodes must let all the other nodes in the network know that they have chosen this role for the current round. To do this, each cluster-head node broadcasts an advertisement message (ADV) using a non-persistent carrier-sense multiple access (CSMA) MAC protocol [69]. This message is a small message containing the node's ID and a header that distinguishes this message as an announcement message. However, this message must be broadcast to reach all of the nodes in the network. There are two reasons for this. First, ensuring that all nodes hear the advertisement essentially eliminates collisions when CSMA is used, since there is no hidden terminal problem (as discussed in Section 2.1.2). Second, since there is no guarantee that the nodes that elect themselves to be cluster-heads are spread evenly throughout the network, using enough power to reach all nodes ensures that every node can become part of a cluster. If the power of the advertisement messages was reduced, some nodes on the edge of the network may not receive any announcements and therefore may not be able to participate in this round of the protocol. Furthermore, since these advertisement messages are small, the increased power to reach all nodes in the network is not a burden. Therefore, the transmit power is set high enough that all nodes within the network can hear the advertisement message.

Each non-cluster-head node determines to which cluster it belongs by choosing the cluster-head that requires the minimum communication energy, based on the *received signal strength* of the advertisement from each cluster-head. Assuming symmetric propagation channels for pure signal strength[1], the cluster-head advertisement heard with the largest signal strength is the cluster-head to whom the minimum amount of transmitted energy is needed for communication. Note that

[1]In the absence of mobile objects moving into or out of the channel, the pure signal strength attenuation of a message sent from a transmitter to a receiver will be the same as the attenuation of a message sent from the receiver to the transmitter because the electromagnetic wave traverses the same path in both cases.

typically this will be the cluster-head closest to the sensor. However, if there is some obstacle impeding the communication between two physically close nodes (e.g., a building, a tree, etc.) such that communication with another cluster-head, located further away, is easier, the sensor will choose the cluster-head that is spatially further away but "closer" in a communication sense. In the case of ties, a random cluster-head is chosen.

After each node has decided to which cluster it belongs, it must inform the cluster-head node that it will be a member of the cluster. Each node transmits a join-request message (Join-REQ) back to the chosen cluster-head using a non-persistent CSMA MAC protocol. This message is again a short message, consisting of the node's ID, the cluster-head's ID, and a header. Since the node has an idea of the relative power needed to reach the cluster-head (based on the received power of the advertisement message), it could adjust its transmit power to this level. However, this approach suffers from the hidden-terminal problem; if a node close to the cluster-head is currently transmitting a join-request message using low transmit power, the remaining nodes in the cluster cannot sense that this transmission is occurring and may decide to transmit their own join-request messages. Since these messages are small, it is more energy-efficient to just increase the transmit power of the join-request messages than to use an 802.11 approach of transmitting request-to-send and clear-to-send (RTS-CTS) messages [24]. This is because the cluster-head does not know the size of its cluster and would need to transmit the CTS message using large power to reach all potential cluster members. In addition, simply increasing the transmit power reduces the latency and increases the sleep time allowed for all the nodes compared to an RTS-CTS approach. Therefore, the nodes use a large power for transmissions of the short join-request messages to the cluster-heads.

The cluster-heads in LEACH act as local control centers to coordinate the data transmissions in their cluster. The cluster-head node sets up a TDMA schedule and transmits this schedule to the nodes in the cluster. This ensures that there are no collisions among data messages and also allows the radio components of each non-cluster-head node to be turned off at all times except during their transmit time, thus minimizing the energy dissipated by the individual sensors. After the TDMA schedule is known by all nodes in the cluster, the set-up phase is complete and the steady-state operation (data transmission) can begin.

A flow-graph of this distributed cluster formation algorithm is shown in Figure 3-3. Figure 3-4 shows an example of the clusters formed during two different rounds of LEACH.

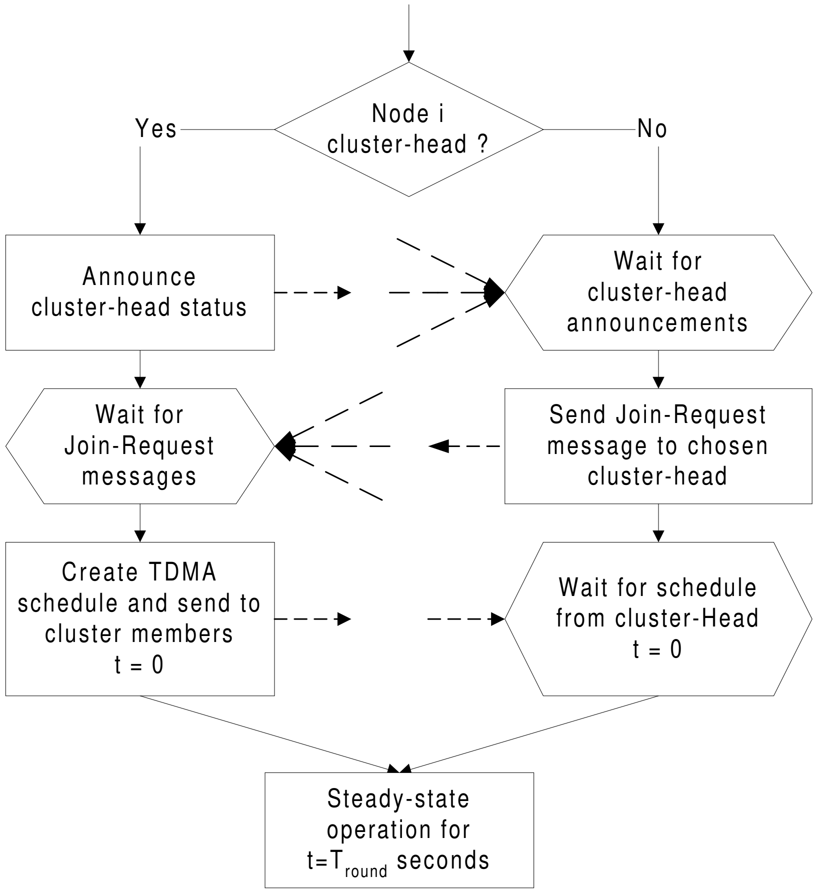


Figure 3-3: Flow-graph of the distributed cluster formation algorithm for LEACH.

3.2 Steady-state Phase

The steady-state operation is broken into frames (see Figure 3-2), where nodes send their data to the cluster-head at most once per frame during their allocated transmission slot. Each slot in which a node transmits data is constant, so the time for a frame of data transfer depends on the number of nodes in the cluster. While the distributed algorithm for determining cluster-head nodes ensures that the expected number of clusters per round is k, it does not guarantee that there are k clusters at each round. In addition, the set-up protocol does not guarantee that nodes are evenly distributed among the cluster-head nodes. Therefore, the number of nodes per cluster is highly variable in LEACH, and the amount of data each node can send to the cluster-head varies depending on the number of nodes in the cluster.

To reduce energy dissipation, each non-cluster-head node uses power control to set the amount of transmit power based on the received strength of the cluster-head advertisement. Furthermore,

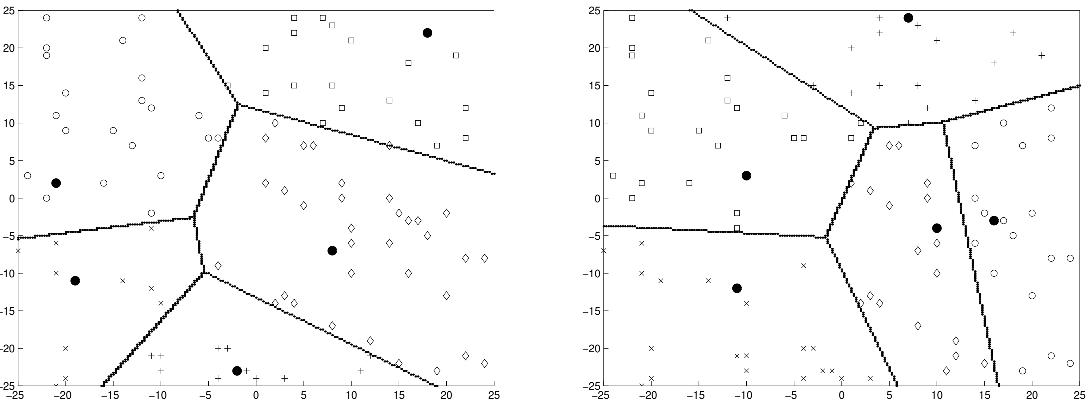


Figure 3-4: Dynamic cluster formation during two different rounds of LEACH. All nodes marked with a given symbol belong to the same cluster, and the cluster-head nodes are marked with •.

the radio of each non-cluster-head node is turned off until its allocated transmission time. Since all the nodes have data to send to the cluster-head and the total bandwidth is fixed, using a TDMA schedule is efficient use of bandwidth and represents a low latency approach, in addition to being energy-efficient.

The cluster-head must keep its receiver on to receive all the data from the nodes in the cluster. Once the cluster-head receives all the data, it can operate on the data (e.g., performing data aggregation, discussed in Section 3.3), and then the resultant data are sent from the cluster-head to the base station. Since the base station may be far away and the data messages are large, this is a high-energy transmission. Figure 3-5 shows a flow-graph of the steady-state operation.

Figure 3-6 shows the time-line for a single round of LEACH, from the time clusters are formed during the set-up phase, through the steady-state operation when data are transfered from the nodes to the cluster-heads and forwarded to the base station.

The preceding discussion describes communication within a cluster. The MAC and routing protocols were designed to ensure low energy dissipation in the nodes and no collisions of data messages within a cluster. However, radio is inherently a broadcast medium. As such, transmission in one cluster will affect (and often degrade) communication in a nearby cluster. For example, Figure 3-7 shows the range of communication for a radio, where node A's transmission, while intended for node B, collides with and corrupts any concurrent transmission intended for node C. To reduce inter-cluster interference, each cluster in LEACH communicates using direct-sequence spread spectrum (DS-SS) (described in Section 2.1.2). Each cluster uses a unique spreading code;

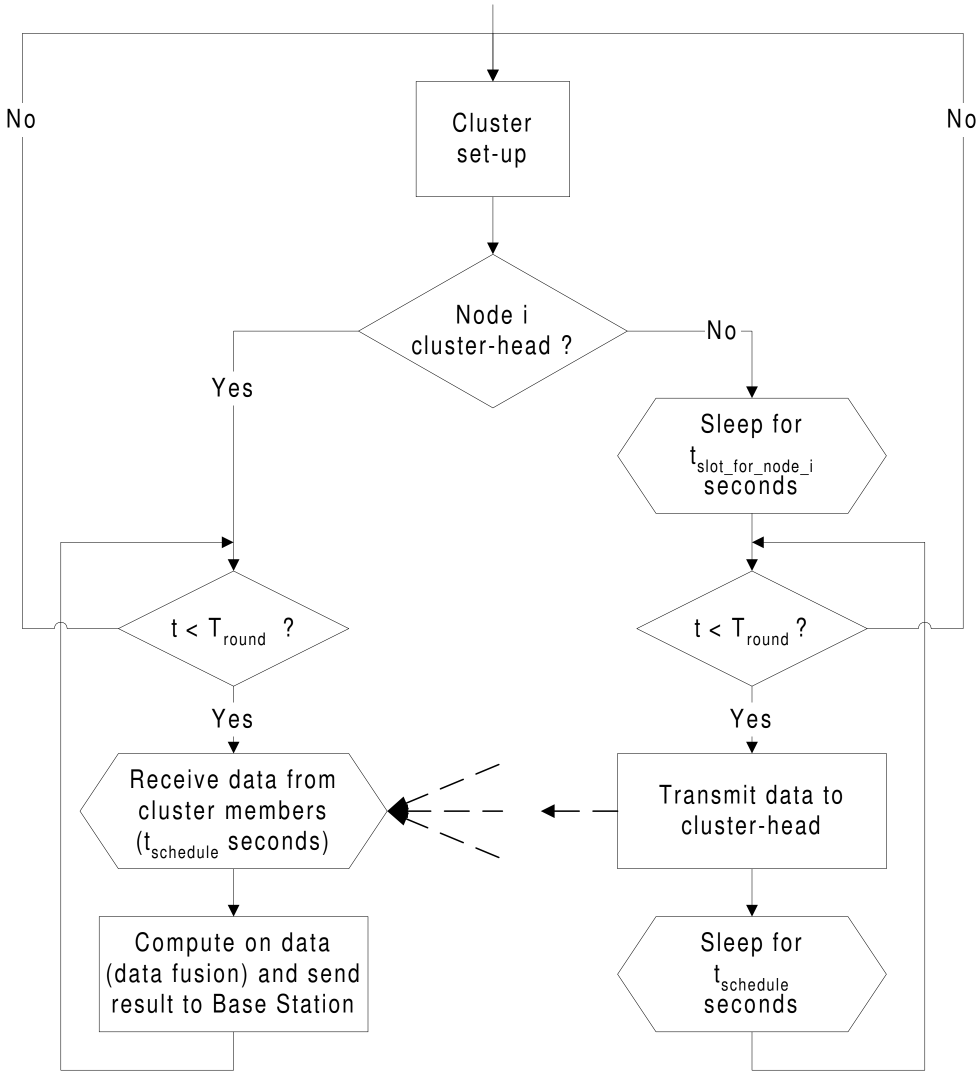


Figure 3-5: Flow-graph of the steady-state operation for LEACH.

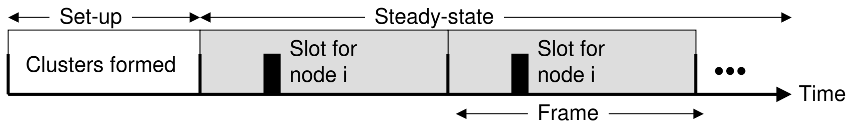


Figure 3-6: Time-line showing LEACH operation. Data transmissions are explicitly scheduled to avoid collisions and increase the amount of time each non-cluster-head node can remain in the sleep state.

all the nodes in the cluster transmit their data to the cluster-head using this spreading code and the cluster-head filters all received energy using this spreading code. This is known as *transmitter-based code assignment* [44], since all transmitters within the cluster use the same code. The first cluster-head to advertise its position is assigned the first code on a pre-defined list, the second cluster-head to advertise its position is assigned the second code, etc[2]. With enough spreading, neighboring clusters' radio signals will be filtered out as noise during de-correlation and not corrupt the transmission from nodes in the cluster. To reduce the possibility of interfering with nearby clusters and reduce its own energy dissipation, each node adjusts its transmit power. Therefore, there will be few overlapping transmissions and little spreading of the data is actually needed to ensure a low probability of collision. Note that each cluster-head only needs a single matched-filter correlator since all the signals destined for it use the same spreading code. This differs from a CDMA approach where each node would have a unique code and the base station receiver would need a bank of matched filters to obtain the data. Combining DS-SS ideas with a TDMA schedule reduces inter-cluster interference while eliminating intra-cluster interference and requiring only a single matched-filter correlator for receiving the data.

Data from the cluster-head nodes to the base station is sent using a fixed spreading code and a CSMA approach. When a cluster-head has data to send (at the end of its frame), it must sense the channel to see if anyone else is transmitting using the base station spreading code. If so, the cluster-head waits to transmit the data. Otherwise, the cluster-head sends the data using the base station spreading code.

3.3 Sensor Data Aggregation

Section 2.2 described the need to aggregate sensor data to produce a meaningful description of events that are occurring in the environment. Data aggregation can be performed on all the unprocessed data at the base station, or it can be performed locally at the cluster-heads. If the energy for communication is greater than the energy for computation, performing data aggregation locally at the cluster-head can reduce the overall system energy consumption, since much less data needs to be transmitted to the base station. This will allow large computation versus communication energy gains with little to no loss in overall network quality.

[2] If there are more clusters than spreading codes, some clusters will use the same code, possibly causing data collisions if the clusters are located close to each other.

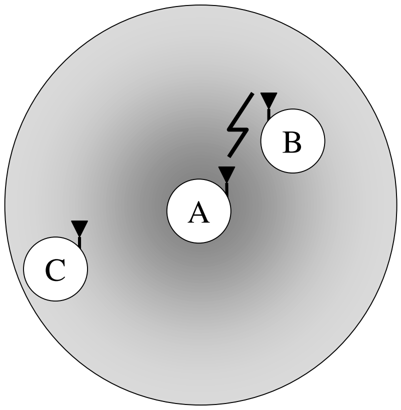


Figure 3-7: Interaction between multiple clusters. Since radio is a broadcast medium, node A's transmission, while intended for node B, collides with and corrupts any concurrent transmission intended for node C.

We can analytically compare the energy dissipation required to perform local data aggregation and send the aggregate data to the base station versus sending the unprocessed data to the base station. Suppose that the energy dissipation per bit for data aggregation is E_{DA} and the energy dissipation per bit to transmit to the base station is E_{TX}. In addition, suppose that the data aggregation method can compress the data with a ratio of L:1. This means that for every L bits that must be sent to the base station when no data aggregation is performed, only 1 bit must be sent to the base station when local data aggregation is performed. Therefore, the energy to perform local data aggregation and transmit the aggregate signal for every L bits of data is:

$$E_{Local-DA} = LE_{DA} + E_{TX} \tag{3.11}$$

and the energy to transmit all L bits of data directly to the base station is:

$$E_{No-DA} = LE_{TX} \tag{3.12}$$

Therefore, performing local data aggregation requires less energy than sending all the unprocessed data to the base station when:

$$\begin{aligned} E_{Local-DA} &< E_{No-DA} \\ LE_{DA} + E_{TX} &< LE_{TX} \\ E_{DA} &< \frac{L-1}{L} E_{TX} \end{aligned} \tag{3.13}$$

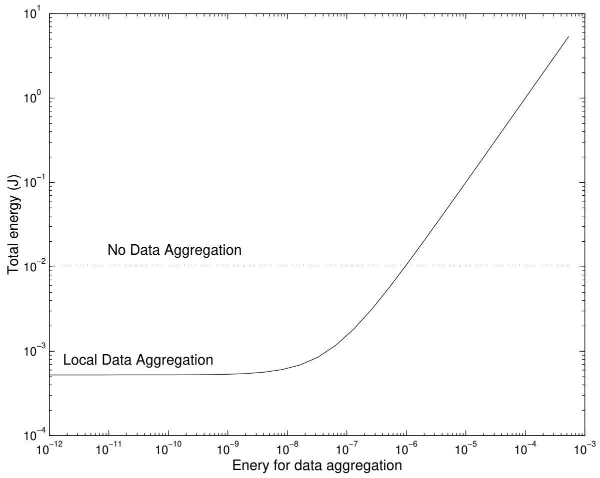


Figure 3-8: Energy dissipation to perform local data aggregation and transmit the aggregate signal to a remote base station compared with sending all the data directly to the base station as the energy cost of performing data aggregation is varied between 1 pJ/bit/signal and 1 mJ/bit/signal.

To confirm these results, we ran an experiment where $N = 20$ nodes sent data to the cluster-head and the data are either aggregated locally, requiring the cluster-head to only send a single signal to the base station ($L = 20$), or all of the unprocessed data are sent to the base station. For this simulation, the base station was 100 m away from the cluster-head node, and the cost for communicating a single bit to the base station was 1.05×10^{-6} J. Figure 3-8 shows the total energy dissipated in the system when local processing is performed and the aggregate data set is sent to the base station (labeled "Local Data Aggregation") versus the total energy dissipated in the system when the unprocessed data signals are sent to the base station (labeled "No Data Aggregation") as the energy required to perform the data aggregation functions varies between 1 pJ/bit/signal and 1 mJ/bit/signal. As expected, when the energy to perform data aggregation is less than $\frac{19}{20}1.05 \times 10^{-6} \approx 1 \times 10^{-6}$ J, the total energy dissipated in the system is less using local processing of the data. However, when the cost of aggregating the signals is higher than 1 μJ/bit/signal, it is more energy-efficient to send the data directly to the base station[3]. This computation versus communication trade-off can be made at the cluster-head node, based on models for the energy-dissipation of computation and communication.

[3]The computation model we use in our simulations, describe in Chapter 4, assumes beamforming data aggregation that consumes 5 nJ/bit/signal.

3.4 Data Correlation

In order for the cluster-head to perform data aggregation to compress the data into a single signal, data from the different nodes in the cluster must be correlated. The question we need to answer is what is the probability that an event will fall into the "view" of all the sensors in the cluster? In other words, how often can we expect to aggregate all signals from cluster members into a single signal that describes the event seen by all the nodes? Alternatively, if we aggregate all data into a single signal, what is the probability that we will miss events?

It is important to have a data-independent model for estimating the amount of correlation that exists between the data from different sensor nodes in order to estimate the amount of compression LEACH can achieve using local data aggregation. If we assume that the source signal travels a distance ρ before it can no longer be reliably detected by the sensors (due to signal attenuation), and that the sensors are omnidirectional (e.g., acoustic, seismic sensors), the maximum distance between sensors with correlated data is 2ρ, as shown in Figure 3-9a. However, being within 2ρ of each other does not guarantee that the two sensors will detect the same signal (Figure 3-9b). If all nodes are within a cluster of diameter d (i.e., the maximum distance between two nodes is d) and $d < 2\rho$, the views of the individual sensors will overlap. This implies that there will be correlation among the data from different sensors in this case. To determine the amount of correlation, we first need to find the percentage of area overlapped by j sensors in order to calculate the probability that a source is detected by j sensors within the cluster. This function f will depend on the parameters ρ and d as well as the total number of nodes in the cluster, N, and is defined as:

$$f(j, N, \rho, d) = \frac{A(j)}{A_{total}} \tag{3.14}$$

where $A(j)$ is the area overlapped by j and only j sensors' views and $A_{total} = \sum_{j=1}^{N} A(j)$ is the total area seen by at least one sensor in the cluster.

Geometrically, it is only possible to bound the area overlapped by all N sensors within a cluster of diameter d. Figure 3-10 shows the *minimum* amount of overlap, which occurs when all sensors are on the circumference of the cluster boundary and $N \to \infty$. From this figure, we see that the total area of overlap of all N nodes in the cluster, $A(N)$, in their views of source signals is:

$$A(N) = \pi(\rho - \frac{d}{2})^2 \tag{3.15}$$

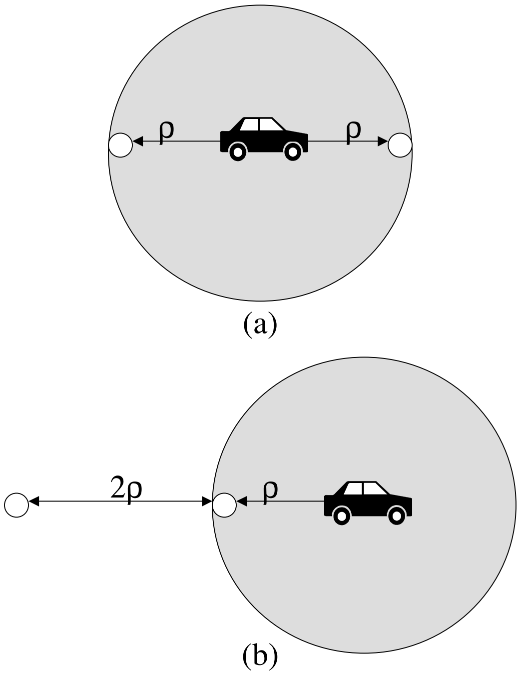


Figure 3-9: Correlation among data sensed at the nodes. If a source signal travels a distance ρ, the maximum distance between correlated data signals is 2ρ (a). However, nodes can be 2ρ apart and have uncorrelated data (b).

The total view seen by each node has area:

$$A_{total} = \pi(\rho + \frac{d}{2})^2 \tag{3.16}$$

Therefore, the fraction of overlap is

$$\begin{aligned} f(j = N, N \to \infty, \rho, d) &= \frac{A(N)}{A_{total}} && (3.17) \\ &= \frac{\pi(\rho - \frac{d}{2})^2}{\pi(\rho + \frac{d}{2})^2} && (3.18) \end{aligned}$$

If d is written as a fraction x of ρ, $d = x\rho$, then the amount of overlap simplifies to

$$f(j = N, N \to \infty, x) = (\frac{1 - \frac{x}{2}}{1 + \frac{x}{2}})^2 \tag{3.19}$$

Figure 3-11 shows the *maximum* amount of overlap, which occurs when $N = 2$, since the addition of more nodes within the cluster of diameter d will reduce the overlap area while increasing the

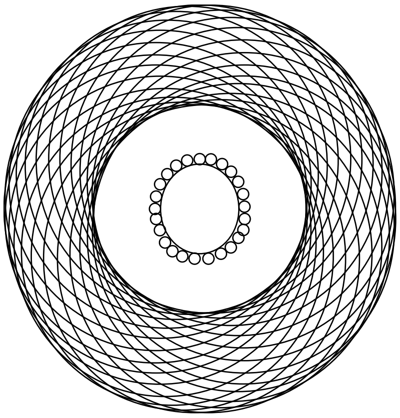


Figure 3-10: If the cluster diameter is $d = x\rho$ and each nodes' view of the world has radius ρ, then the fraction of area seen by all N nodes in the cluster is minimized when $N \to \infty$ and all N nodes are on the cluster circumference. In this case, $f(j = N, N \to \infty, x) = \frac{(1-\frac{x}{2})^2}{(1+\frac{x}{2})^2}$.

total area. From this diagram, we can geometrically find the area of overlap:

$$A_{\frac{1}{2}} = \frac{\theta}{\pi}\pi\rho^2 - 2(\frac{1}{2}\frac{d}{2}\sqrt{\rho^2 - \frac{d^2}{4}}) \tag{3.20}$$

Since $\theta = \cos^{-1}\frac{d}{2\rho}$ and $d = x\rho$, this simplifies to:

$$A_{\frac{1}{2}} = \rho^2 \cos^{-1}\frac{x}{2} - \frac{x\rho^2}{2}\sqrt{1 - \frac{x^2}{4}} \tag{3.21}$$

The total area of overlap of the N nodes is $A(N) = 2A_{\frac{1}{2}}$ and the total area covered by the two nodes is $A_{total} = 2(\pi\rho^2) - A(N)$. Therefore, the fraction of overlap is:

$$f(j = N, N = 2, x) = \frac{A(N)}{A_{total}} \tag{3.22}$$

$$= \frac{2\cos^{-1}\frac{x}{2} - x\sqrt{1 - \frac{x^2}{4}}}{2\pi - 2\cos^{-1}\frac{x}{2} + x\sqrt{1 - \frac{x^2}{4}}} \tag{3.23}$$

As mentioned previously, d must be less than 2ρ ($x < 2$) for there to be overlap in the views of all the sensors. Figure 3-12 shows the lower bound $f(j = N, N \to \infty, x)$ (Equation 3.19) and the upper bound $f(j = N, N = 2, x)$ (Equation 3.23) for $0 < x < 2$. From this figure, we see that $f(j = N, N, x)$ will depend greatly on the value of x. If x is small (i.e., the distance between the nodes is much smaller than their views of the world ρ), $f(j = N, N, x)$ is large, whereas if x is

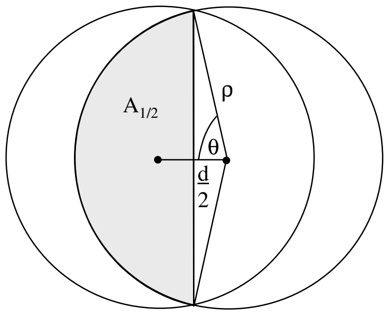


Figure 3-11: If the cluster diameter is $d = x\rho$ and each nodes' view of the world has radius ρ, then the fraction of area seen by all N nodes in the cluster is maximized when $N = 2$. In this case, $f(j = N, N = 2, x) = \frac{2\cos^{-1}\frac{x}{2} - x\sqrt{1-\frac{x^2}{4}}}{2\pi - 2\cos^{-1}\frac{x}{2} + x\sqrt{1-\frac{x^2}{4}}}$.

greater than 1 (i.e., $d > \rho$), there is very little area seen by every node. In addition, $f(j = N, N, x)$ depends on the value of N; if N is small, $f(j = N, N, x)$ will be closer to the upper bound, and if N is large, it will be closer to the lower bound.

3.5 LEACH-C: Base Station Cluster Formation

The previous sections described LEACH in detail. While there are advantageous to using LEACH's distributed cluster formation algorithm, where each node makes autonomous decisions that result in all the nodes being placed into clusters, this protocol offers no guarantee about the placement and/or number of cluster-head nodes. Since the clusters are adaptive, obtaining a poor clustering set-up during a given round will not greatly affect overall performance of LEACH. However, using a central control algorithm to form the clusters may produce better clusters by dispersing the cluster-head nodes throughout the network. This is the basis for LEACH-C (LEACH-Centralized), a protocol that uses a centralized clustering algorithm and the same steady-state protocol as LEACH (e.g., nodes send their data to the cluster-head, and the cluster-head aggregates the data and sends the aggregate signal to the base station).

During the set-up phase of LEACH-C, each node sends information about its current location and energy level to the base station. The base station runs an optimization algorithm to determine the clusters for that round. The clusters formed by the base station will in general be better

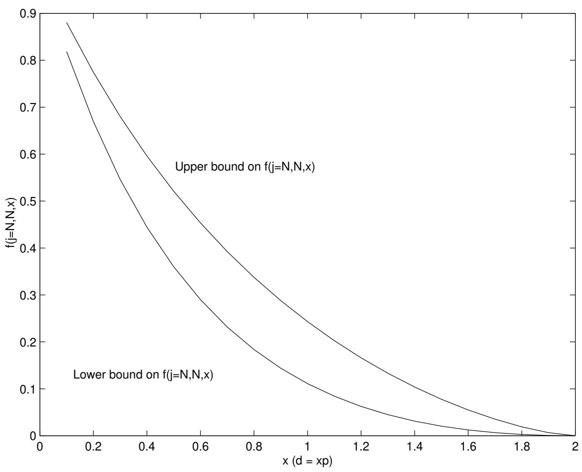


Figure 3-12: Upper and lower bounds for $f(j = N, N, x)$, where N is the number of nodes in the network, as a function of x, where $d = x\rho$ is the diameter of the cluster and ρ is the radius of each node's view of the environment.

than those formed using the distributed algorithm. However, LEACH-C requires that each node transmit information about its location to the base station at the beginning of each round. This information may be obtained by using a global positioning system (GPS) receiver that is activated at the beginning of each round to get the node's current location [61].

Determining optimal clusters from the nodes is a problem that is known to be NP-Hard [4]. Approximation algorithms, such as taboo search or simulated annealing [66], can be used to approach the optimal solution in polynomial time. Simulated annealing is an algorithm based on thermodynamics principles. If a solid material is melted and allowed to cool, the energy of the system enters several intermediate states before settling at the low-energy final state. If the system enters a state that is lower in energy than its previous state, the system remains there. However, if the system enters a state that is higher in energy than its previous state, the system remains there with a probability given by:

$$p = e^{-\frac{\Delta E}{k_{Boltz} T}} \tag{3.24}$$

where k_{Boltz} is the Boltzmann constant and T is a fixed temperature. This algorithm can be applied to optimization problems where ΔE is replaced with the difference in cost between the new state and the old state, and $k_{Boltz}T$ is a parameter that must be picked to ensure that the algorithm converges.

In addition to determining good clusters, the base station needs to ensure that the energy load is evenly distributed among all the nodes. To do this, the base station computes the average node energy, and whichever nodes have energy below this average cannot be cluster-heads for the current round. Using the remaining nodes as possible cluster-heads, the base station runs a simulated annealing algorithm to determine the best k nodes to be cluster-heads for the next round and the associated clusters. This algorithm minimizes the amount of energy the non-cluster-head nodes will have to use to transmit their data to the cluster-head, by minimizing the total sum of squared distances between all the non-cluster-head nodes and the closest cluster-head[4]. At each iteration, the next state, which consists of a set of nodes in $C\prime$, is determined from the current state, the set of nodes in C, by randomly (and independently) perturbing the x and y coordinates of the nodes c in C to get new coordinates $x\prime$ and $y\prime$. The nodes that have location closest to ($x\prime$,$y\prime$) become the new set of cluster-head nodes $c\prime$ that make up set $C\prime$. Given the current state at iteration k, represented by the set of cluster-head nodes C with cost $f(C)$, the new state, represented by the set of cluster-head nodes $C\prime$ with cost $f(C\prime)$, will become the current state with probability:

$$p_k = \begin{cases} e^{-(f(C\prime)-f(C))/\alpha_k} & : \quad f(C\prime) \geq f(C) \\ 1 & : \quad f(C\prime) < f(C) \end{cases}$$

where α_k is the control parameter (equivalent to the temperature parameter in the thermodynamic model) and $f(\cdot)$ represents the cost function defined by

$$f(C) = \sum_{i=1}^{N} \min_{c \in C} d^2(i,c) \tag{3.25}$$

where $d(i,c)$ is the distance between node i and node c. The parameter α_k must be chosen to be increasing with increasing k to ensure that the algorithm converges. However, if α_k increases too quickly, the system will get stuck in local minima. On the other hand, if α_k increases too slowly, the system will take a very long time to converge. Using simulations, we found that the following value for α_k works well for determining good clusters:

$$\alpha_k = 1000e^{\frac{k}{20}} \tag{3.26}$$

[4]As noted previously, communication energy may not scale exactly with distance, e.g., if a building or tree is impeding a good communication channel. However, gathering information about the communication channel between all nodes is impractical. Using distance is therefore an approximation of the amount of energy that will be required for communication.

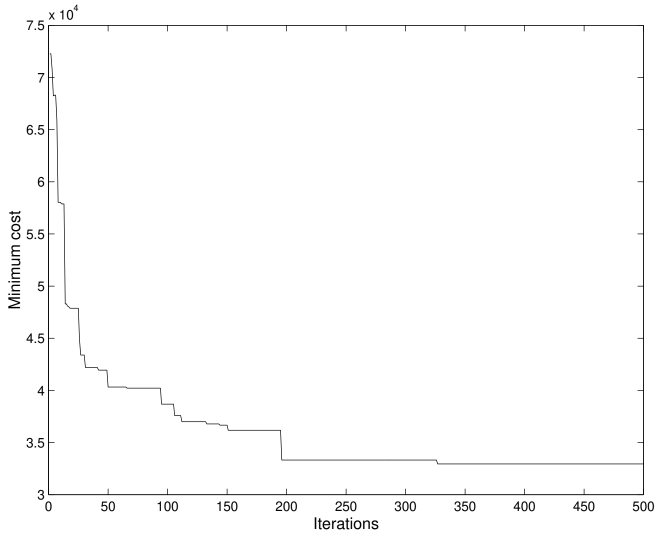


Figure 3-13: Cost function as the simulated annealing algorithm progresses. The algorithm typically converges in 200-500 iterations for a 100 node network.

Using this value of α_k, the algorithm typically converges in 200-500 iterations for a 100 node network. Figure 3-13 shows the overall decrease in the cost function as the algorithm progresses. Since these computations are being performed at the base station, energy dissipation is not a concern.

Once the optimal cluster-heads and associated clusters are found, the base station transmits this information back to all the nodes in the network. This is done by broadcasting a message that contains the cluster-head ID for each node. If a node's cluster-head ID matches its own ID, that node takes on the cluster-head role; otherwise, the node determines its TDMA slot for data transmission and goes to sleep until it is time to transmit data to its cluster-head. The steady-state phase of LEACH-C is identical to that of LEACH.

3.6 LEACH-F: Fixed Cluster, Rotating Cluster-Head

Adapting the clusters depending on which nodes are cluster-heads for a particular round is advantageous because it ensures that nodes communicate with the cluster-head node that requires the lowest amount of transmit power. In addition to reducing energy dissipation, this ensures minimum inter-cluster interference, as the power of an interfering message will be less than (or, at most, equal to) the power of the message the cluster-head is receiving (see Figure 3-14). If, on

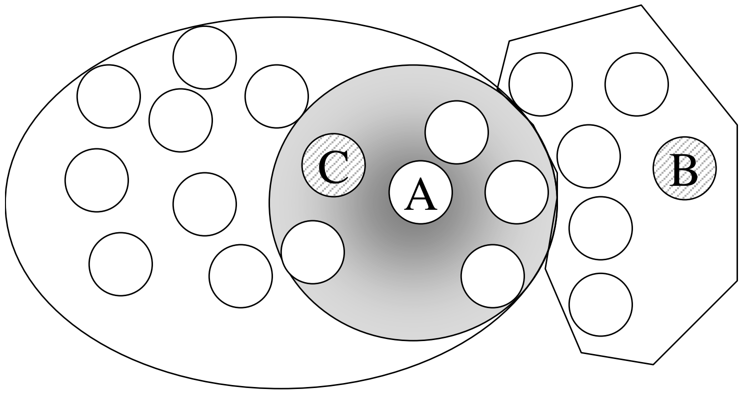


Figure 3-14: If the clusters are adaptive and change depending on the location of the cluster-head nodes, there is only minimal inter-cluster interference. In this figure, node A chooses to join cluster-C because it requires less transmit power to communicate with node C than node B, the other choice for cluster-head. In addition to minimizing the non-cluster-head nodes' energy dissipation, adaptive clustering reduces inter-cluster interference.

the other hand, the clusters were fixed and only the cluster-head nodes were rotated, a node may have to use a large amount of power to communicate with its cluster-head when there is another cluster's cluster-head close by. For example, in Figure 3-15, node A needs to use a large amount of transmit power to communicate with its cluster-head, node B. Since cluster-head C is located close to node A, node A's transmission will corrupt any transmission to cluster-head C. Therefore, using fixed clusters and rotating cluster-head nodes within the cluster may require more transmit power from the nodes, increasing non-cluster-head node energy dissipation and increasing inter-cluster interference.

The advantage of fixed clusters is that once the clusters are formed, there is no set-up overhead at the beginning of each round. Depending on the cost for forming adaptive clusters, an approach where the clusters are formed once and fixed and the cluster-head position rotates among the nodes in the cluster may be more energy-efficient than LEACH. This is the basis for LEACH-F (LEACH with Fixed clusters). In LEACH-F, clusters are created using the centralized cluster formation algorithm developed for LEACH-C. The base station uses simulated annealing to determine optimal clusters and broadcasts the cluster information to the nodes. This broadcast message includes the cluster ID for each node, from which the nodes can determine the TDMA schedule and the order to rotate the cluster-head position. The first node listed in the cluster becomes cluster-head for the first round, the second node listed in the cluster becomes cluster-head for the second round, and

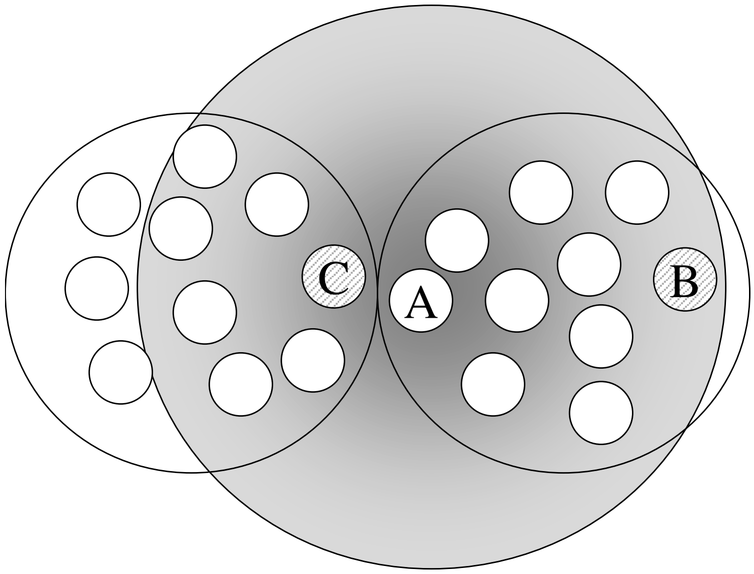


Figure 3-15: If the clusters are fixed and only the cluster-head nodes are rotated, there can be significant inter-cluster interference. In this figure, node A has to use a large amount of transmit power to communicate with its cluster-head, node B. Since cluster-head C is much closer to node A than cluster-head B, node A's transmission will cause a large amount of interference to any transmissions cluster-head C is receiving from its cluster members.

so forth. Using LEACH-F, there is no set-up required for the different rounds—nodes implicitly know when they are cluster-heads and when they are non-cluster-heads[5]. The steady-state phase of LEACH-F is identical to that of LEACH.

LEACH-F would not be practical in any sort of dynamic system. The fixed nature of this protocol does not allow new nodes to be added to the system and does not adjust its behavior based on nodes dying. Furthermore, LEACH-F does not handle node mobility. Therefore, while this is a good comparison protocol to determine the advantage of a no-overhead approach, it may not be a useful protocol architecture for real systems.

3.7 Summary

This chapter introduced our cross-layer protocol architecture, LEACH. LEACH was designed to exploit the application-specific function of sensor networks, where the end-user requires information about events occurring in the environment, rather than nodes' individual data. In addition, LEACH

[5]In a practical system, there would probably be some set-up at the beginning of each round to ensure the nodes in the cluster are all time-synchronized.

was designed to enable maximum energy savings by enabling nodes to enter the sleep state, where portions of the node are powered-down to save energy, as often as possible.

Since nodes located close to each other often have highly correlated data, we would like these nodes to be able to share data to enable local processing. This suggests grouping the nodes into local clusters. However, conventional clustering approaches require high-powered cluster-head nodes. In order to meet the ease of deployment design goal outlined in Section 1.1.1, we do not want to require that a high-powered cluster-head node infrastructure exists. Since we assume that all nodes are energy-limited, we introduced the idea of rotating the cluster-head position among all the nodes in the network. To ensure a minimum amount of energy dissipation in the non-cluster-head nodes, we use adaptive clustering, where non-cluster-head nodes can join the cluster which allows the easiest communication to the cluster-head node. Using adaptive clusters and rotating cluster-head nodes adds reliability to the system as well, since no assumptions are made about the state of nodes in the network.

We have developed a distributed cluster formation algorithm. This algorithm allows individual nodes to make decisions without knowledge of the decisions being made by the other nodes in the network but produces clusters that satisfy design criterion (e.g., that there be k clusters, on average, and that each node share equally in the load of being a cluster-head). Such a distributed approach is fault-tolerant, as nodes are not reliant on specific other nodes in the network and there are no nodes that are more important than other nodes.

We developed a centralized cluster formation algorithm as part of the LEACH-C protocol. Using this protocol requires that each node know its location in order to generate a topology map. Furthermore, the nodes must send this information to a remote base station. However, once the base station has the information, it can create better clusters from the nodes than can be achieved using a purely distributed approach. In addition, this set-up algorithm guarantees k clusters per round. We have also discussed the LEACH-F protocol. In this protocol, the clusters are created initially and fixed throughout the system lifetime; the cluster-head position is rotated within these fixed clusters. This reduces overhead but makes the system too inflexible to be used in a any sort of dynamic network.

By implementing a cross-layer design, LEACH is better suited to the functions of wireless microsensor networks. LEACH uses application-specific knowledge to meet the design goals; performing local data processing (e.g., data aggregation) will increase system lifetime (since energy dissipation is reduced) and latency (since less data is being transmitted throughout the network).

At the same time, quality will be minimally affected, since the data aggregation processing would be done at the base station as well. The next chapter will show quantitatively how well LEACH performs compared to general-purpose approaches to data routing.

Chapter 4

Analysis and Simulation of LEACH

For even moderately-sized networks with tens of nodes, it is impossible to analytically model the interactions between all the nodes. Therefore, simulation was used to determine the benefits of different protocols. Computation and communication energy dissipation models as well as new MAC algorithms were implemented in `ns` to support the design and simulation of the different protocol architectures. In the experiments described in this chapter, LEACH is compared with LEACH-C (the centralized set-up algorithm), LEACH-F (the fixed cluster, rotating cluster-head algorithm), MTE routing (where data traverses multiple short hops to reach the base station), and static clustering (where clusters and cluster-head nodes are fixed) in terms of system lifetime, energy dissipation, amount of data transfer (actual data for MTE routing, aggregate data for the LEACH protocols), and latency.

4.1 Simulation Models

In order to compare different protocols, it is important to have good models for all aspects of communication. This section describes the models that were used for channel propagation, communication energy dissipation, and computation energy dissipation.

4.1.1 Channel Propagation Model

In a wireless channel, the electromagnetic wave propagation can be modeled as falling off as a power law function of the distance between the transmitter and receiver. In addition, if there is no direct, line-of-sight path between the transmitter and the receiver, the electromagnetic wave will bounce off objects in the environment and arrive at the receiver from different paths at different times.

This causes multipath fading, which again can be roughly modeled as a power law function of the distance between the transmitter and receiver. No matter which model is used (direct line-of-sight or multipath fading), the received power decreases as the distance between the transmitter and receiver increases [75].

For the experiments described in this dissertation, both the free space model and the multipath fading model were used, depending on the distance between the transmitter and receiver, as defined by the channel propagation model in `ns` [14, 75]. If the distance between the transmitter and receiver is less than a certain cross-over distance ($d_{crossover}$), the Friss free space model is used (d^2 attenuation), and if the distance is greater than $d_{crossover}$, the two-ray ground propagation model is used (d^4 attenuation). The cross-over point is defined as follows:

$$d_{crossover} = \frac{4\pi\sqrt{L}h_r h_t}{\lambda} \tag{4.1}$$

where

$L \geq 1$ is the system loss factor not related to propagation,

h_r is the height of the receiving antenna above ground,

h_t is the height of the transmitting antenna above ground, and

λ is the wavelength of the carrier signal.

If the distance is less than $d_{crossover}$, the transmit power is attenuated according to the Friss free space equation as follows:

$$P_r(d) = \frac{P_t G_t G_r \lambda^2}{(4\pi d)^2 L} \tag{4.2}$$

where

$P_r(d)$ is the receive power given a transmitter-receiver separation of d,

P_t is the transmit power,

G_t is the gain of the transmitting antenna,

G_r is the gain of the receiving antenna,

λ is the wavelength of the carrier signal,

d is the distance between the transmitter and the receiver, and

$L \geq 1$ is the system loss factor not related to propagation.

This equation models the attenuation when the transmitter and receiver have direct, line-of-sight communication, which will only occur if the transmitter and receiver are close to each other (i.e., $d < d_{crossover}$). If the distance is greater than $d_{crossover}$, the transmit power is attenuated according to the two-ray ground propagation equation as follows:

$$P_r(d) = \frac{P_t G_t G_r h_t^2 h_r^2}{d^4} \tag{4.3}$$

where

$P_r(d)$ is the receive power given a transmitter-receiver separation of d,

P_t is the transmit power,

G_t is the gain of the transmitting antenna,

G_r is the gain of the receiving antenna,

h_r is the height of the receiving antenna above ground,

h_t is the height of the transmitting antenna above ground, and

d is the distance between the transmitter and the receiver.

In this case, the received signal comes from both the direct path and a ground-reflection path [75]. Due to destructive interference when there is more than one path through which the signal arrives, the signal is attenuated as d^4.

In the experiments described in this dissertation, an omnidirectional antenna was used with the following parameters: $G_t = G_r = 1$, $h_t = h_r = 1.5$ m, no system loss ($L = 1$), 914 MHz radios, and $\lambda = \frac{3\times10^8}{914\times10^6} = 0.328$ m. Using these values, $d_{crossover} = 86.2$ m and Equations 4.2 and 4.3 simplify to:

$$P_r = \begin{cases} 6.82\times10^{-4}\frac{P_t}{d^2} & : \quad d < 86.2 \text{ m} \\ 2.25\frac{P_t}{d^4} & : \quad d \geq 86.2 \text{ m} \end{cases} \tag{4.4}$$

4.1.2 Radio Energy Model

There has been a significant amount of research in the area of low-energy radios. Different assumptions about the radio characteristics, including energy dissipation in the transmit and receive modes, will change the advantages of different protocols. In this work, we assume a simple model where the transmitter dissipates energy to run the radio electronics and the power amplifier and the receiver dissipates energy to run the radio electronics [88]. As discussed in the previous section,

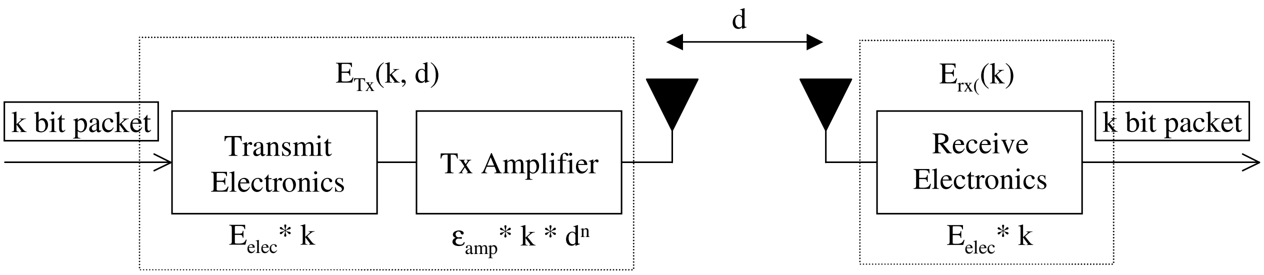


Figure 4-1: Radio energy dissipation model.

the power attenuation is dependent on the distance between the transmitter and receiver. For relatively short distances, the propagation loss can be modeled as inversely proportional to d^2, whereas for longer distances, the propagation loss can be modeled as inversely proportional to d^4. Power control can be used to invert this loss by setting the power amplifier to ensure a certain power at the receiver. Thus, to transmit an l-bit message a distance d, the radio expends:

$$E_{Tx}(l,d) = E_{Tx-elec}(l) + E_{Tx-amp}(l,d) \tag{4.5}$$

$$E_{Tx}(l,d) = \begin{cases} lE_{elec} + l\epsilon_{friss-amp}d^2 & : \quad d < d_{crossover} \\ lE_{elec} + l\epsilon_{two-ray-amp}d^4 & : \quad d \geq d_{crossover} \end{cases} \tag{4.6}$$

and to receive this message, the radio expends:

$$\begin{aligned} E_{Rx}(l) &= E_{Rx-elec}(l) \\ E_{Rx}(l) &= lE_{elec} \end{aligned} \tag{4.7}$$

as shown in Figure 4-1. The electronics energy, E_{elec} depends on factors such as the digital coding, modulation, and filtering of the signal before it is sent to the transmit amplifier. In addition, when using DS-SS, the electronics energy accounts for the spreading of the data when transmitting and the correlation of the data with the spreading code when receiving. Researchers have designed transceiver baseband chips that support multi-user spread-spectrum communication and operate at 165 mW in transmit mode and 46.5 mW in receive mode [83]. For the experiments described in this dissertation, we set the energy dissipated per bit in the transceiver electronics to be

$$E_{elec} = 50 \text{ nJ/bit} \tag{4.8}$$

for a 1 Mbps transceiver. This means the radio electronics dissipates 50 mW when in operation (either transmit or receiver)[1].

The parameters $\epsilon_{friss-amp}$ and $\epsilon_{two-ray-amp}$ will depend on the required receiver sensitivity and the receiver noise figure, as the transmit power needs to be adjusted so that the power at the receiver is above a certain threshold, $P_{r-thresh}$. We can work backwards from this receive power threshold to determine the minimum transmit power. If the radio bitrate is R_b, the transmit power, P_t is equal to the transmit energy per bit $E_{Tx-amp}(1,d)$ times the bitrate:

$$P_t = E_{Tx-amp}(1,d)R_b \tag{4.9}$$

Plugging in the value of $E_{Tx-amp}(1,d)$ gives:

$$P_t = \begin{cases} \epsilon_{friss-amp}R_b d^2 & : \quad d < d_{crossover} \\ \epsilon_{two-ray-amp}R_b d^4 & : \quad d \geq d_{crossover} \end{cases} \tag{4.10}$$

Using the channel models described in the previous section, the received power is:

$$P_r = \begin{cases} \frac{\epsilon_{friss-amp}R_b G_t G_r \lambda^2}{(4\pi)^2} & : \quad d < d_{crossover} \\ \epsilon_{two-ray-amp}R_b G_t G_r h_t^2 h_r^2 & : \quad d \geq d_{crossover} \end{cases} \tag{4.11}$$

The parameters $\epsilon_{friss-amp}$ and $\epsilon_{two-ray-amp}$ can be determined by setting Equation 4.11 equal to $P_{r-thresh}$:

$$\epsilon_{friss-amp} = \frac{P_{r-thresh}(4\pi)^2}{R_b G_t G_r \lambda^2} \tag{4.12}$$

$$\epsilon_{two-ray-amp} = \frac{P_{r-thresh}}{R_b G_t G_r h_t^2 h_r^2} \tag{4.13}$$

Therefore, the required transmit power, P_t, as a function of the receiver threshold and the distance between the transmitter and receiver is:

$$P_t = \begin{cases} \alpha_1 P_{r-thresh} d^2 & : \quad d < d_{crossover} \\ \alpha_2 P_{r-thresh} d^4 & : \quad d \geq d_{crossover} \end{cases} \tag{4.14}$$

where $\alpha_1 = \frac{(4\pi)^2}{G_t G_r \lambda^2}$ and $\alpha_2 = \frac{1}{G_t G_r h_t^2 h_r^2}$.

[1]When the data is spread, as in the steady-state phase of LEACH, the transmitter and receiver are on for longer than when the data is not spread. Therefore, sending and receiving a spread-spectrum signal requires more electronics energy than sending and receiving a non-spread signal.

We can determine the receiver threshold $P_{r-thresh}$ using estimates for the noise at the receiver. If the thermal noise floor is 99 dBm [18] and the receiver noise figure is 17 dB[2], and we require a signal-to-noise ratio (SNR) of at least 30 dB to receive the signal with no errors, the minimum receive power $P_{r-thresh}$ for successful reception is

$$P_{r-thresh} \geq 30 + (-82) = -52 \text{ dBm} \tag{4.15}$$

Therefore, the received power must be at least -52 dBm or 6.3 nW for successful reception of the packet. Plugging the values that will be used in the experiments ($G_t = G_r = 1$, $h_t = h_r = 1.5$ m, $\lambda = 0.328$ m, and $R_b = 1$ Mbps) into Equations 4.12 and 4.13 gives:

$$\epsilon_{friss-amp} = 10 \text{ pJ/bit/m}^2 \tag{4.16}$$

$$\epsilon_{two-ray-amp} = 0.0013 \text{ pJ/bit/m}^4 \tag{4.17}$$

These are the radio energy parameters that will be used for the simulations described in this chapter.

4.1.3 Beamforming Energy Model

The results of experiments described in [94] were used to model the computational costs of performing beamforming data aggregation. Alice Wang ran experiments implementing the least mean square (LMS) and Maximum Power beamforming algorithms (see Section 2.2.2) on a StrongARM processor and measured the energy dissipation. Figure 4-2 shows the results of these experiments. This figure shows that the LMS beamforming algorithm requires much less energy than the Maximum Power beamforming algorithm. In addition, the energy for LMS beamforming scales linearly with the number of sensors, while the energy for Maximum Power beamforming scales quadratically with the number of sensors. Therefore, the LMS beamforming algorithm is better-suited for implementation on a low-power microsensor node. Figure 4-2 shows that implementing the LMS beamforming algorithm on the SA-1100 requires 5 μJ/sample/signal, or 625 nJ/bit/signal. It is reasonable to assume that there would be 1-2 orders of magnitude reduction in energy dissipation if the beamforming was implemented using an application-specific integrated circuit (ASIC) (as was shown for an encryption algorithm in [33]). Therefore, computation energy for beamforming BF

[2]Note that according to [75], the noise figure for an AMPS cellular phone is -119.5 dBm, so a noise figure of 10 dB + (-99 dBm) = -82 dBm is reasonable.

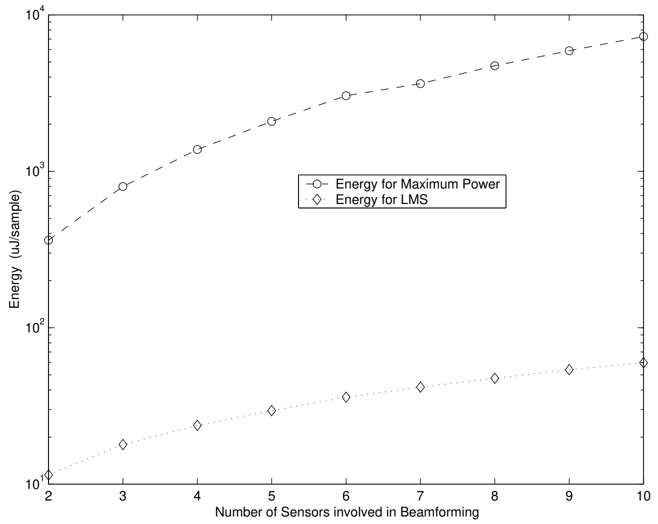


Figure 4-2: Energy dissipated using the StrongARM-1100 (SA-1100) to implement the LMS and Maximum Power beamforming algorithms. This plots shows that there is a linear relationship between the LMS beamforming algorithm and the number of sensors whereas there is a quadratic relationship between the Maximum Power beamforming algorithm and the number of sensors.

is set to 5 nJ/bit/signal.

These equation for modeling the computation and communication energy dissipation and the associated parameters are summarized in Tables 4.1.

4.1.4 `ns` Extensions

To implement LEACH and the general-purpose comparison protocols, we added several features to `ns` [67], an event-driven network simulator with extensive support for simulation of wireless network protocols. The extensions include MAC protocols, energy dissipation models for computation and communication, and the protocol architectures discussed in this dissertation (LEACH, LEACH-C, LEACH-F, MTE routing, and static clustering). Detailed discussion of these extensions to `ns` can be found in Appendix A.

4.2 Experimental Set-up

We ran wireless microsensor network simulations using `ns` to determine the benefits of the different protocol architectures discussed in this dissertation. For these experiments, the random, 100-node network shown in Figure 4-3 was used. The base station was placed 75 meters from the closest

Table 4.1: Radio characteristics and parameter values.

Description	Parameter	Value
Cross-over distance for Friss and two-ray ground attenuation models	$d_{crossover}$	$\frac{4\pi h_r h_t}{\lambda}$
Transmit power	P_t	$\epsilon_{friss-amp} R_b d^2 \quad : \quad d < d_{crossover}$ $\epsilon_{two-ray-amp} R_b d^4 \quad : \quad d \geq d_{crossover}$
Receive power	P_r	$\frac{\epsilon_{friss-amp} R_b G_t G_r \lambda^2}{(4\pi)^2} \quad : \quad d < d_{crossover}$ $\epsilon_{two-ray-amp} R_b G_t G_r h_t^2 h_r^2 \quad : \quad d \geq d_{crossover}$
Minimum receiver power needed for successful reception	$P_{r-thresh}$	6.3 nW
Radio amplifier energy	$\epsilon_{friss-amp}$ $\epsilon_{two-ray-amp}$	$\frac{P_{r-thresh}(4\pi)^2}{R_b G_t G_r \lambda^2}$ $\frac{P_{r-thresh}}{R_b G_t G_r h_t^2 h_r^2}$
Radio electronics energy	E_{elec}	50 nJ/bit
Compute energy for beamforming	E_{BF}	5 nJ/bit
Bitrate	R_b	1 Mbps
Antenna gain factor	G_t, G_r	1
Antenna height above the ground	h_t, h_r	1.5 m
Signal wavelength	λ	0.325 m
Cross-over distance for Friss and two-ray ground attenuation models	$d_{crossover}$	87 m
Radio amplifier energy	$\epsilon_{friss-amp}$ $\epsilon_{two-ray-amp}$	10 pJ/bit/m^2 0.0013pJ/bit/m^4

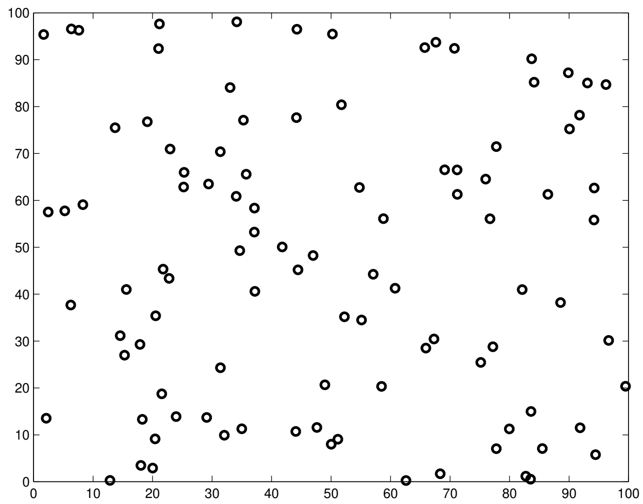


Figure 4-3: 100-node random test network. The base station is located 75 meters from the closest node, at location (x=50, y=175) (not shown).

Table 4.2: Characteristics of the test network.

Nodes	100
Network size	100 m $\times$ 100 m
Base station location	(50, 175)
Radio propagation speed	3x10^8 m/s
Processing delay	50 μs
Radio speed	1 Mbps
Data size	500 bytes

node, at location (x=50, y=175) (not shown in the figure). The bandwidth of the channel was set to 1 Mbps, and the processing delay was 25 μs on the transmitting side and 25 μs on the receiving side. Each data message was 500 bytes long, and the packet header for each type of packet was 25 bytes long. The radio electronics energy was set to 50 nJ/bit and the radio transmitter energy was set to 10 pJ/bit/m^2 for distances less than 87 m and 0.0013 pJ/bit/m^4 for distances greater than 87 m. The energy for performing beamforming computations to aggregate data was set to 5 nJ/bit/signal. These parameters are summarized in Tables 4.1 and 4.2.

4.3 Optimum Number of Clusters

In LEACH, the cluster formation algorithm was created to ensure that the expected number of clusters per round is k, a system parameter. What is the optimum value of k that will minimize energy dissipation in the system? We can analytically determine the optimal number of clusters in a LEACH system using the computation and communication energy models developed in the previous section. Assume there are N nodes distributed uniformly in an $M \times M$ region. If there are k clusters, there are on average $\frac{N}{k}$ nodes per cluster. Each cluster-head dissipates energy receiving signals from the nodes, beamforming the signals, and transmitting the aggregate signal to the base station. Since the base station is located away from the nodes, presumably the distance to the base station is greater than the cross-over distance and the energy dissipation follows the two-ray ground model (e.g., d^4 power loss). Therefore, the energy dissipated in the cluster-head node during a single frame is:

$$E_{CH} = lE_{elec}\frac{N}{k} + lE_{BF}\frac{N}{k} + l\epsilon_{two-ray-amp}d^4_{toBS} \tag{4.18}$$

where l is the number of bits in each data message and d_{toBS} is the distance from the cluster-head node to the base station.

Each non-cluster-head node only needs to transmit its data to the cluster-head once during a frame. Presumably the distance between the non-cluster-head node and its cluster-head is less than the cross-over distance, so the energy dissipation follows the Friss free-space model (e.g., d^2 power loss). Thus, the energy used in each non-cluster-head node is:

$$E_{non-CH} = lE_{elec} + l\epsilon_{friss-amp}d^2_{toCH} \tag{4.19}$$

where d_{toCH} is the distance from the node to the cluster-head. The area occupied by each cluster is approximately $\frac{M^2}{k}$. In general, this is an arbitrary-shaped region with a node distribution $\rho(x, y)$. The expected squared distance from the nodes to the cluster-head (assumed to be at the center of mass of the cluster) is given by:

$$\begin{aligned} E[d^2_{toCH}] &= \int\int (x^2 + y^2)\rho(x, y)dxdy \\ &= \int\int r^2 \rho(r, \theta) r dr d\theta \end{aligned} \tag{4.20}$$

If we assume this area is a circle with radius $R = \frac{M}{\sqrt{\pi k}}$ and $\rho(r, \theta)$ is constant for r and θ, Equation 4.20 simplifies to:

$$
\begin{aligned}
E[d_{toCH}^2] &= \rho \int_{\theta=0}^{2\pi} \int_{r=0}^{\frac{M}{\sqrt{\pi k}}} r^3 dr d\theta \\
&= \frac{\rho}{2\pi} \frac{M^4}{k^2}
\end{aligned} \tag{4.21}
$$

If the density of nodes is uniform throughout the cluster area, then $\rho = \frac{1}{\frac{M^2}{k}}$ and

$$
E[d_{toCH}^2] = \frac{1}{2\pi} \frac{M^2}{k} \tag{4.22}
$$

Therefore, in this case,

$$
E_{non-CH} = lE_{elec} + l\epsilon_{friss-amp} \frac{1}{2\pi} \frac{M^2}{k} \tag{4.23}
$$

The energy dissipated in a cluster during the frame is:

$$
E_{cluster} = E_{CH} + \frac{N}{k} E_{non-CH} \tag{4.24}
$$

and the total energy for the frame is:

$$
\begin{aligned}
E_{total} &= kE_{cluster} \\
&= l(E_{elec}N + E_{BF}N + k\epsilon_{two-ray-amp} d_{toBS}^4 + NE_{elec} + N\epsilon_{friss-amp} \frac{1}{2\pi} \frac{M^2}{k})
\end{aligned} \tag{4.25}
$$

We can find the optimum number of clusters by setting the derivative of E_{total} with respect to k to zero:

$$
\begin{aligned}
\frac{dE_{total}}{dk} &= 0 \\
\epsilon_{two-ray-amp} d_{toBS}^4 &= N\epsilon_{friss-amp} (\frac{1}{2\pi} \frac{M^2}{k^2}) \\
k &= \frac{\sqrt{N}}{\sqrt{2\pi}} \sqrt{\frac{\epsilon_{friss-amp}}{\epsilon_{two-ray-amp}}} \frac{M}{d_{toBS}^2}
\end{aligned} \tag{4.26}
$$

For our experiments, $N = 100$ nodes, $M = 100$ m, $\epsilon_{friss-amp} = 10$ pJ, $\epsilon_{two-ray-amp} = 0.0013$ pJ, and $75 < d_{toBS} < 185$, so we expect the optimum number of clusters to be:

$$
1 < k < 6 \tag{4.27}
$$

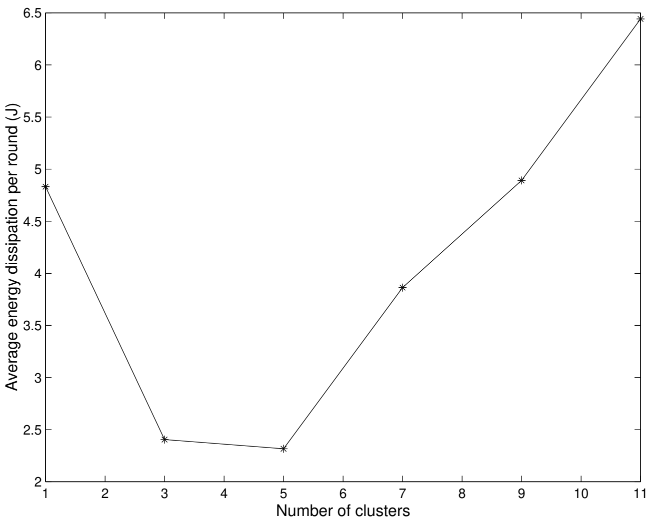


Figure 4-4: Average energy dissipated per round in LEACH as the number of clusters is varied between 1 and 11. This graph shows that LEACH is most energy-efficient when there are between 3 and 5 clusters in the 100-node network, as predicted by the analysis.

These analytical results were verified using simulations where we varied the number of clusters between 1 and 11 and ran LEACH for 1000 simulated seconds. Figure 4-4 shows the average energy dissipated per round as a function of the number of clusters. This graph shows that the optimum number of clusters is, as predicted by the analysis, around 3-5 for the 100-node network. When there is only 1 cluster, the non-cluster-head nodes often have to transmit data very far to reach the cluster-head node, draining their energy, and when there are more than 5 clusters, there is not as much local data aggregation being performed. For the rest of the experiments, the parameter k will be set to 5.

4.4 How Often to Rotate Cluster-Heads?

There is a cost in terms of time and energy to set up the clusters for LEACH. Therefore, the steady-state phase should be long compared with the set-up phase to amortize the overhead of cluster formation. On the other hand, if the energy at each node is limited, running the steady-state phase for too long will drain the energy of the cluster-head node and curtail communication between the non-cluster-head nodes that have energy and the base station. Therefore, there is a trade-off in how long to make the steady-state phase. Equations 4.18 and 4.23 describe the energy usage of a cluster-head and a non-cluster-head node during a single frame of data transfer. The total energy drained from each node per round depends on the average number of frames per round, $N_{frames/round}$, as follows:

$$E_{CH/round} = N_{frames/round} \times E_{CH/frame} \tag{4.28}$$

$$E_{non-CH/round} = N_{frames/round} \times E_{non-CH/frame} \tag{4.29}$$

where $E_{CH/frame}$ is the energy to receive all signals from non-cluster-head nodes, the energy to aggregate the signals, and the energy to send the aggregate data to the base station, while $E_{non-CH/frame}$ is the energy to send a data signal to the cluster-head.

One method of determining how often to rotate the clusters is to ensure that each node's energy lasts long enough to allow the node to be a cluster-head once during its lifetime and a non-cluster-head node during the other rounds of the network operation. If there are $\frac{N}{k}$ rounds before each node has been a cluster-head, this means each node should have enough energy to be a cluster-head once and a non-cluster-head $(\frac{N}{k}-1)$ times. Combining Equations 4.18 and 4.23 with Equations 4.28 and 4.29 and assuming that each node begins with E_{start} Joules of energy gives the following:

$$\begin{aligned} E_{CH/round} + (\frac{N}{k} - 1)E_{non-CH/round} &= E_{start} \\ N_{frames/round}(lE_{elec}\frac{N}{k} + lE_{BF}\frac{N}{k} + l\epsilon_{two-ray-amp}d_{toBS}^4) + & \\ N_{frames/round}(\frac{N}{k} - 1)(lE_{elec} + l\epsilon_{friss-amp}\frac{1}{2\pi}\frac{M^2}{k}) &= E_{start} \end{aligned} \tag{4.30}$$

Therefore, we can solve for $N_{frames/round}$ as a function of the system parameters:

$$N_{frames/round} = \frac{E_{start}/l}{[(E_{elec}\frac{N}{k} + E_{BF}\frac{N}{k} + \epsilon_{two-ray-amp}d_{toBS}^4) + (\frac{N}{k} - 1)(E_{elec} + \epsilon_{friss-amp}\frac{1}{2\pi}\frac{M^2}{k})]} \tag{4.31}$$

Assuming there are $\frac{N}{k}$ nodes per cluster and each l-bit data message takes $t_{msg} = \frac{l}{R_b}$ seconds, the total frame time is $t_{frame} = \frac{N}{k}\frac{l}{R_b}$ seconds. The time for each round is thus:

$$\begin{aligned} t_{round} &= N_{frames/round} \times t_{frame} \\ &= \frac{1}{R_b}\frac{N}{k}\frac{E_{start}}{[(E_{elec}\frac{N}{k} + E_{BF}\frac{N}{k} + \epsilon_{two-ray-amp}d_{toBS}^4) + (\frac{N}{k} - 1)(E_{elec} + \epsilon_{friss-amp}\frac{1}{2\pi}\frac{M^2}{k})} \end{aligned} \tag{4.32}$$

Plugging in the values E_{elec} = 50 nJ/bit, E_{BF} = 5 nJ/bit, $\epsilon_{friss-amp}$ = 10 pJ/bit/m^2, $\epsilon_{two-ray-amp}$ = 0.0013 pJ/bit/m^4, $N = 100$, $M = 100$ m, $k = 5$, $l = 4000$ bits, and $R_b = 1$ Mbps bits gives:

$$
\begin{aligned}
N_{frames/round} &= \frac{E_{start}}{9 \text{ mJ}} \\
t_{msg} &= 4 \text{ ms} \\
t_{frame} &= 80 \text{ ms} \\
t_{round} &= 0.08 \text{ seconds} * \frac{E_{start}}{9 \text{ mJ}}
\end{aligned}
\tag{4.33}
$$

Therefore, given the initial energy of the nodes, E_{start}, the cluster-heads and associated clusters should be rotated approximately every $0.08\frac{E_{start}}{0.009}$ seconds.

4.5 Simulation Results

For the experiments described in this section, we implemented LEACH, LEACH-C, LEACH-F, MTE routing, and static clustering. We will briefly summarize each of these protocols.

In LEACH, nodes organize themselves into clusters using the distributed algorithm described in Section 3.1. Once the clusters are formed, the cluster-head nodes create TDMA schedules. Nodes transmit their data during their assigned slot, and the cluster-head aggregates all the data into a representative signal to send to the base station. This protocol has the advantage of being distributed, self-configuring, and not requiring location information for cluster formation. In addition, the steady-state protocol is low-energy. However, the draw-back is that there is no guarantee as to the number or placement of cluster-head nodes within the network.

LEACH-C uses a centralized cluster formation algorithm to guarantee k nodes in the cluster and minimize the total energy spent by the non-cluster-head nodes by evenly distributing the cluster-head nodes throughout the network. The steady-state protocol in LEACH-C is the same as LEACH, where nodes transmit data to the cluster-head, and the cluster-head performs data aggregation to reduce the data sent to the base station. This protocol produces a better cluster distribution than LEACH, as it has global knowledge of the location of all nodes in the network. However, this requires that nodes be equipped with GPS or other location-finding algorithms. In addition, if the base station is very far away from the network, the cost to configure the network will be high.

In LEACH-F, the clusters are formed using a centralized protocol and fixed throughout the network lifetime. The cluster-head nodes are rotated within the fixed clusters to distribute the energy load. While this reduces set-up at the beginning of each round, the non-cluster-head nodes may be required to send their data to a cluster-head node further away than another that belongs to a different cluster. Therefore, depending on the energy cost for set-up of the adaptive clusters and the size of the clusters, LEACH-F may or may not be energy-efficient. In addition, this approach is not flexible to node mobility or nodes being added or removed from the network.

For MTE routing, nodes set up routes where they minimize the amount of transmit energy required to get their data to the base station by using several short hops. The routes are computed at the beginning of the simulation. When a node dies, its upstream neighbors send their data to its next-hop neighbor to keep the network connected. This requires that the upstream neighbors increase their transmit energy. When there is no correlation in the data (and hence data aggregation cannot be performed within the network), this is a good approach to getting data to the base station while minimizing energy dissipation. However, in sensor networks, where there is a large amount of correlation within the network, extra data is needlessly transmitted to the base station.

Finally, static clustering sets up fixed clusters with fixed cluster-head nodes. The nodes use a TDMA schedule to send data to the cluster-head, and the cluster-head aggregates the data before transmission to the base station. This approach has little overhead, but when the cluster-head node runs out of energy, the nodes within the cluster lose communication ability with the base station.

4.5.1 Nodes Begin with Equal Energy

For the first set of experiments, each node begins with only 2 J of energy[3] and an unlimited amount of data to send to the base station. Since all nodes begin with equal energy in these simulations, each node uses the probabilities in Equation 3.2 to determine its cluster-head status at the beginning of each round, and each round lasts for 20 seconds (as per the analysis in the previous section, $t_{round} = \frac{0.08*2}{0.009} \approx 20$ seconds). We tracked the rate at which the data are transfered to the base station and the amount of energy required to get the data to the base station. Since the nodes have limited energy, they use up this energy during the course of the simulation. Once a node runs out of energy, it is considered dead and can no longer transmit or receive data.

For these simulations, energy is removed whenever a node transmits or receives data and when-

[3]Assuming nickel cadmium (NiCd) technology, this corresponds to a 15 mg battery [30].

ever it performs data aggregation. Using spread-spectrum increases the number of bits transmitted, thereby increasing the amount of energy dissipated in the electronics of the radio. Therefore, the energy to transmit or receive a signal depends on whether or not spread-spectrum is being used. We do not assume any static energy dissipation, nor do we remove energy during carrier-sense operations.

Although quality is an application-specific and data-dependent quantity, one application-independent method of determining quality is to measure the amount of data (number of actual data signals or number of data signals represented by an aggregate signal) received at the base station. The more data the base station receives, the more accurate its view of the remote environment will be. If all the nodes within a cluster are sensing the same event, the actual and effective data will contain the same information, and there is no loss in quality by sending effective or aggregate data rather than actual data. If, on the other hand, the nodes are seeing different events, the cluster-head will pick out the strongest event (strongest signal within the signals of the cluster members) and send that as the data from the cluster. In this case, there will be a loss in quality by aggregating signals into a single representative signal. If the distance between nodes within a cluster is small compared with the distance from which events can be sensed or if the distance between events occurring in the environment is large, there is a high probability that the nodes will be sensing the same event.

Figure 4-5 shows the total number of data signals (actual for MTE, effective for LEACH, LEACH-C, LEACH-F, and static clustering) received at the base station over time, the total energy dissipated over time, and the total data received at the base station per given amount of energy. Figure 4-5a shows that LEACH sends much more data to the base station in the simulation time than MTE routing and therefore achieves low latency. The reason MTE requires so much time to send data from the nodes to the base station is that each message traverses several hops. In the other protocols, each message is transmitted over a single hop, to the cluster-head, where data aggregation occurs. The aggregate signals are then sent directly to the base station. Therefore, much less data needs to be sent the long distance to the base station.

Figure 4-5b shows the total energy dissipated over time. LEACH, LEACH-C, LEACH-F, and MTE use up all the energy available in the network (2 J/node $\times$ 100 nodes = 200 J), while static-clustering is unable to take advantage of the energy remaining in the non-cluster-head nodes.

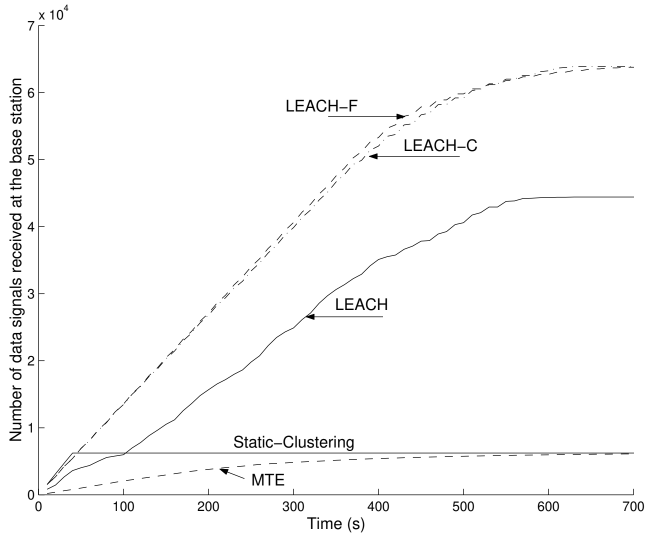
While LEACH and MTE use the same total amount of energy over the simulation time, Figure 4-5a shows that LEACH delivers an order of magnitude more data to the base station than MTE routing. This can be seen clearly in Figure 4-5c, the total data received at the base station per given amount of energy. This graph shows that LEACH, LEACH-C, and LEACH-F deliver the most data per unit energy. Therefore, these protocols are more energy- and latency-efficient than MTE. Using a routing protocol such as MTE does not enable local computation to reduce the amount of data that needs to be transmitted to the base station. In addition to taking a large amount of time to get data from the nodes to the base station, routing protocols require a large amount of energy.

Figure 4-5 shows that LEACH is almost as efficient as LEACH-C (LEACH-C delivers about 40% more data per unit energy than LEACH). This is because the base station has global knowledge of the location and energy of all the nodes in the network, so it can produce better clusters that require less energy for data transmission. In addition, the base station formation algorithm ensures that there are $k = 5$ clusters during each round of operation. As there are only 100 nodes in the simulation, even though the expected number of clusters per round is $k = 5$ in LEACH, each round does not always have 5 clusters. Figure 4-6 shows the distribution of the number of clusters using LEACH for this simulation. While the average is 5, some rounds have as little as 1 cluster and some rounds have as many as 10 clusters. Therefore, the base station algorithm, which always ensures 5 clusters, should perform better than distributed clustering.

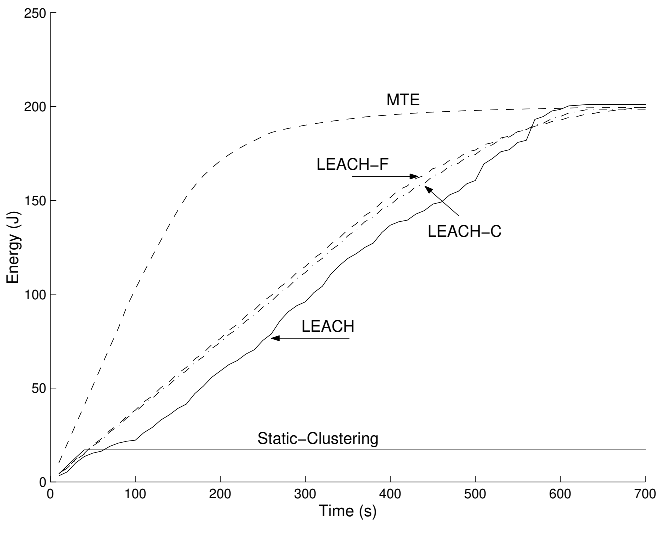
LEACH-F performs as well as LEACH-C for these simulations. The energy saved by having no set-up at the beginning of each round offsets the increase in energy from having nodes sometimes transmit their data to a further cluster-head.

As seen in Figure 4-5, static clustering performs poorly, because the cluster-head nodes die quickly, ending the lifetime of all nodes belonging to those clusters. Figure 4-7a shows the total number of nodes that remain alive over the simulation time. While nodes remain alive for a long time in MTE, this is because a much smaller amount of data has been transmitted to the base station. If we plot the total number of nodes that remain alive per amount of data received at the base station (Figure 4-7b, we see that nodes in LEACH can deliver ten times more effective data than MTE for the same number of node deaths. Therefore, nodes in LEACH are able to better use the available energy.

Figure 4-7 also shows the large advantage in rotating the cluster-head nodes and the associated clusters. In the static clustering approach, where clusters are fixed, as soon as the cluster-head node dies, all the *other* nodes in the cluster are essentially dead, because they have lost communication with the base station. Thus the system lifetime using static clustering is significantly shortened.

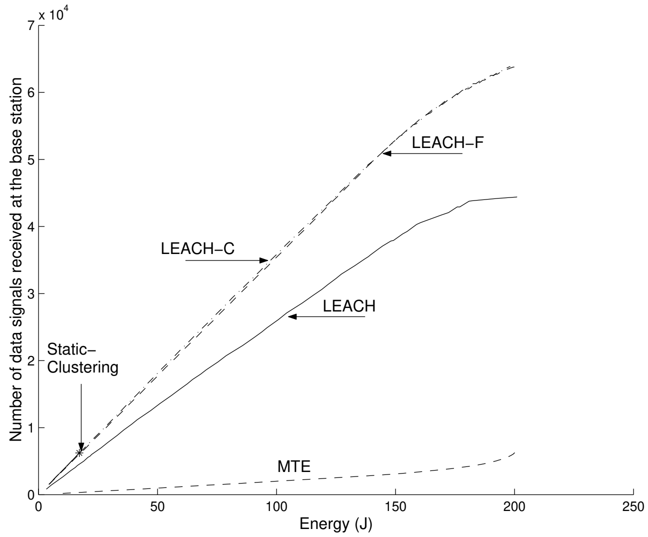


(a)



(b)

Figure 4-5: Data for the limited energy simulations, where each node begins with 2 J of energy. (a) The total amount of data received at the base station over time. (b) The total amount of energy dissipated in the system over time. (Figure continued on the next page.)



(c)

Figure 4-5: (Cont.) Data for the limited energy simulations, where each node begins with 2 J of energy. (c) The total amount of data received at the base station per given amount of energy. These graphs show that LEACH distributes an order of magnitude more data per unit energy than MTE routing, LEACH-C delivers 40% more data per unit energy than LEACH, LEACH-F performs similar to LEACH-C, and static-clustering does not perform well when the nodes have limited energy.

4.5.2 Varying the Base Station Location

The results presented in the previous section show that LEACH is more energy- and latency-efficient than MTE routing. Is this just a function of the simulation parameters? What happens if the base station is actually located within the network or very far away from the network? To answer these questions, we ran simulations where we varied the location of the base station with respect to the network shown in Figure 4-3. Figure 4-8 shows the amount of data per unit energy that each of the protocols delivers to the base station as the location of the base station varies from ($x = 50$, $y = 50$) to ($x = 50$, $y = 300$). From this plot, we see that when the base station is in the center of the network ($y = 50$), LEACH delivers 5 times the amount of data per unit energy as MTE routing, whereas LEACH-C delivers 7 times the amount of data per unit energy. As the base station moves further away from the network, the performance of LEACH improves compared to MTE routing. For all base station locations we simulated, LEACH performs better than MTE routing by at least a factor of 5 and as much as an order of magnitude, whereas LEACH-C performs better than MTE routing by at least a factor of 7 and as much as a factor of 16. Table 4.3 summarizes the performance comparisons.

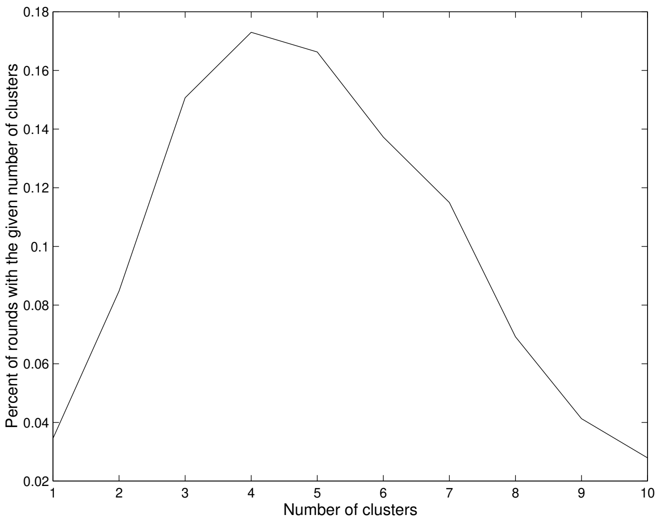
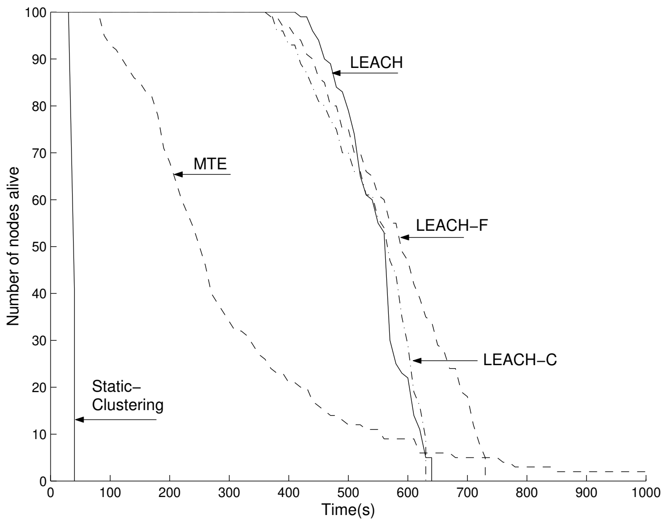


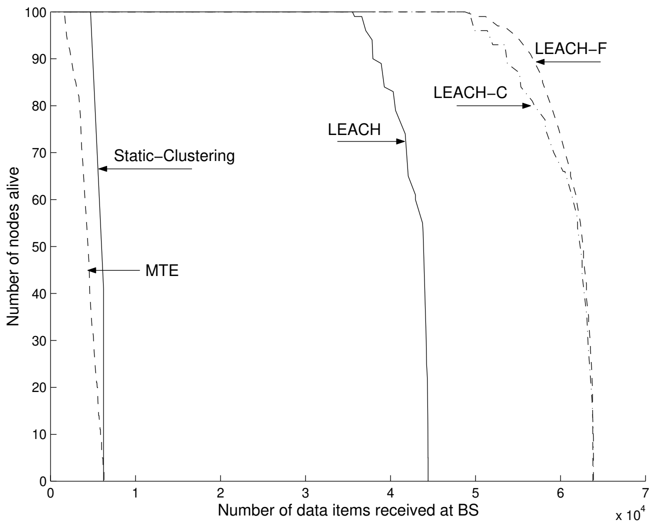
Figure 4-6: Distribution of the number of clusters in each round in LEACH. While each node chooses to be a cluster-head with a probability to ensure $E[\#\ \mathrm{CH}\ ] = 5$, there are only 100 nodes in the network, so occasionally there are as few as 1 cluster and as many as 10 clusters. However, on average, there are 5 clusters in the network.

MTE routing performs significantly better when the base station is located within the network (65% better than when the base station is on the edge of the network and increasing to 250% better when the base station is 250 m from the center of the network). This is because when the base station is located within the network, there is no long-distance hop across which nodes need to send data. This saves a large amount of energy. Since MTE routing has more data to send to the base station, the savings compared with the base station located far from the network are more significant than for LEACH or LEACH-C. However, LEACH is still able to perform at least five times better than MTE routing, even for the case when the base station is at the center of the network.

Static clustering delivers the most data per unit energy of all the protocols, but the total amount of data delivered (and the total system lifetime) is much shorter than with the other approaches, as shown in Figure 4-9. This graph shows the total amount of data received at the base station during the simulation time. As noted previously, static clustering cannot send a large amount of data to the base station because the cluster-head nodes in static clustering use of their limited energy quickly, ending the communication of all the nodes in the cluster.



(a)



(b)

Figure 4-7: Data for the limited energy simulations, where each node begins with 2 J of energy. (a) Number of nodes alive over time. (b) Number of nodes alive per amount of data sent to the base station. LEACH can deliver 10 times the amount of effective data to the base station as MTE routing for the same number of node deaths. The benefit of rotating cluster-heads in LEACH is clearly seen by comparing the number of nodes alive in LEACH and static-clustering.

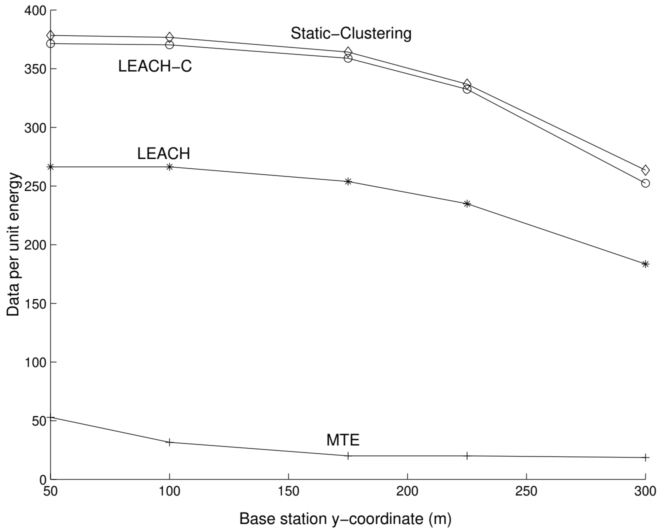


Figure 4-8: Average data per unit energy as the base station location varies between ($x = 50$, $y = 50$) and ($x = 50$, $y = 300$) for the different protocols. Note that ($x = 50$, $y = 50$) represents the middle of the network.

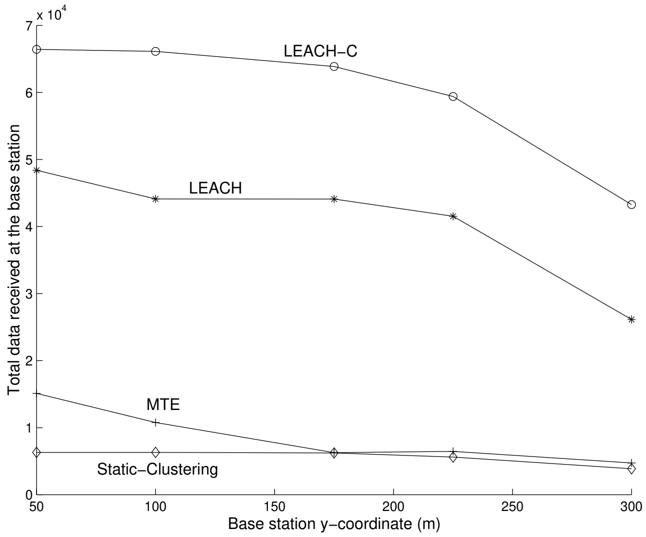


Figure 4-9: Total data received at the base station as the base station location varies between ($x = 50$, $y = 50$) and ($x = 50$, $y = 300$) for the different protocols. This graph shows that even though static clustering has good data per unit energy performance, the total data that can be delivered to the base station is limited by the deaths of the cluster-head nodes.

Table 4.3: Performance of the different protocols as the base station location is varied.

Base Station Location/ Distance From Network Center	Protocol	Data Per Unit Energy	Performance Improvement Over MTE Routing
(x=50, y=50) 0 m	LEACH	266	5.0
	LEACH-C	371	7.0
	MTE	53	1
	Static-Clustering	378	7.1
(x=50, y=100) 50 m	LEACH	266	8.4
	LEACH-C	370	11.7
	MTE	32	1
	Static-Clustering	377	12.0
(x=50, y=175) 125 m	LEACH	254	12.6
	LEACH-C	359	17.9
	MTE	20	1
	Static-Clustering	364	18.1
(x=50, y=225) 175 m	LEACH	235	11.7
	LEACH-C	333	16.5
	MTE	20	1
	Static-Clustering	337	16.8
(x=50, y=300) 250 m	LEACH	184	9.8
	LEACH-C	252	13.5
	MTE	19	1
	Static-Clustering	264	14.1

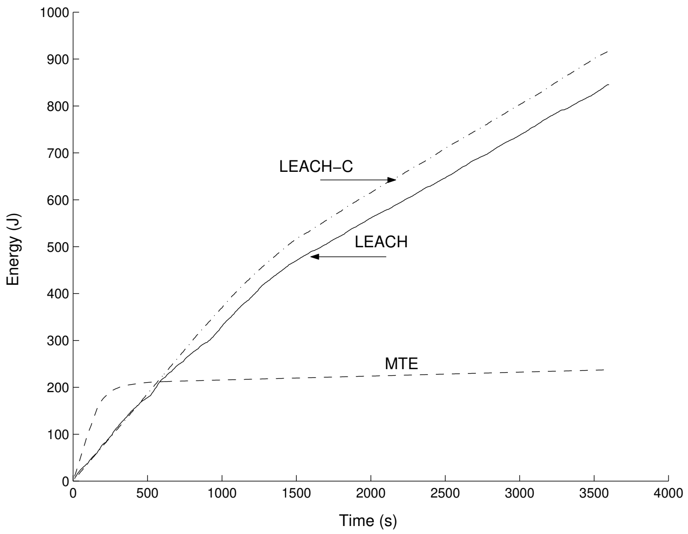
4.5.3 Nodes Begin with Unequal Energy

To see how well LEACH can utilize any high-energy nodes that are in the network, we ran simulations where 10 nodes began with 200 J of energy and the remaining 90 nodes began with only 2 J of energy. Since nodes began with unequal energies, each node used the probabilities in Equation 3.5 to determine its cluster-head status at the beginning of each round[4], and each round lasted 20 seconds to ensure that the limited energy of some nodes would not be drained during the round. Figure 4-10 shows the total data received at the base station over time, the total energy dissipated over time, and the total data received at the base station per given amount of energy for LEACH, LEACH-C, and MTE routing. These graphs show that LEACH is an order of magnitude more energy-efficient than MTE routing.

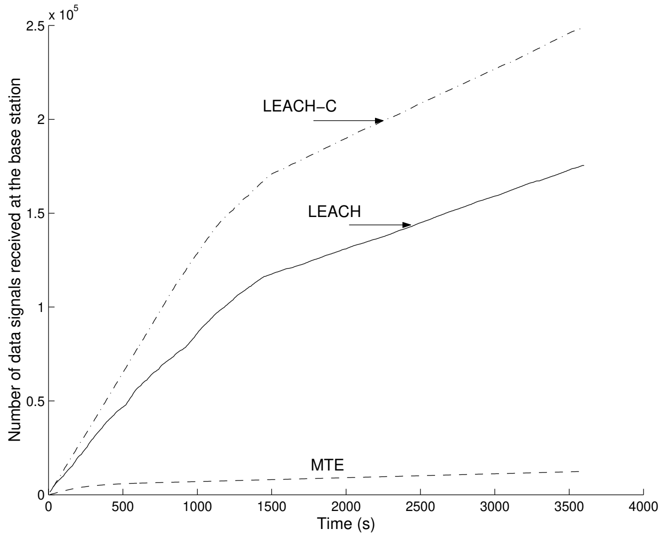
Nodes in MTE routing die early, since the routes can not take advantage of the high-energy nodes. This can be seen in Figure 4-11, the number of nodes that are alive over time and the number of nodes that are alive per data received at the base station. This graph shows that nodes in MTE routing die after delivering only a small amount of data to the base station, whereas nodes in LEACH and LEACH-C, which do take advantage of the high-energy nodes, remain alive to deliver over 50 times the amount of data for the same number of node deaths. A power-aware routing protocol [70, 85] would be able to utilize the high-energy nodes and should greatly increase system lifetime compared to MTE routing.

Figure 4-12 shows the total number of times each node in the network was a cluster-head using LEACH. Each of the 10 nodes that began the simulations with 200 J was a cluster-head an average of 89 times during the simulation, whereas each of the 90 nodes that began the simulations with 2 J was a cluster-head an average 0.25 times during the simulation. Therefore, the 10 high-energy nodes were cluster-heads over 350 times more often than the 90 low-energy nodes. These results show that using the probabilities in Equation 3.5, LEACH is able to take advantage of any high-energy nodes in the network.

[4]For these simulations, there was no cost associated with determining the total energy in the network.

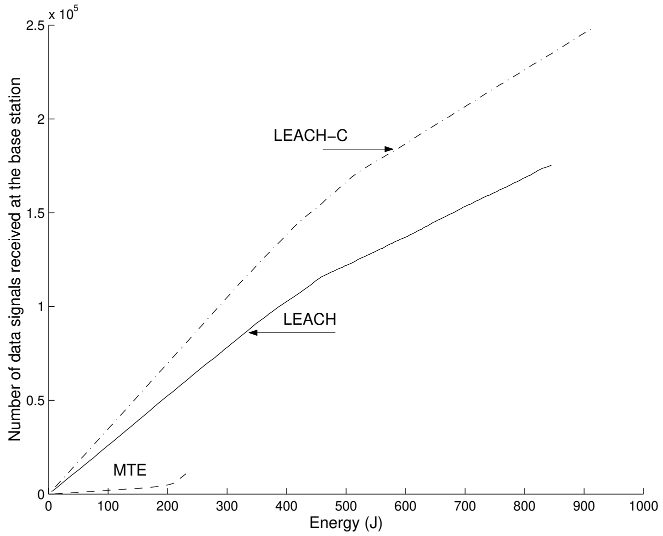


(a)



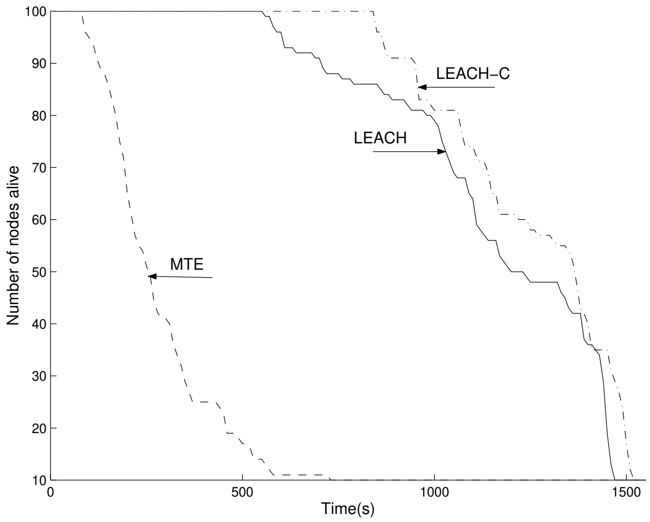
(b)

Figure 4-10: Data for the unequal energy simulations, where 10 nodes began with 200 J of energy and the remaining 90 nodes began the simulations with 2 J of energy. (a) The total amount of data received at the base station over time. (b) The total amount of energy dissipated in the system over time. (Figure continued on the next page.)

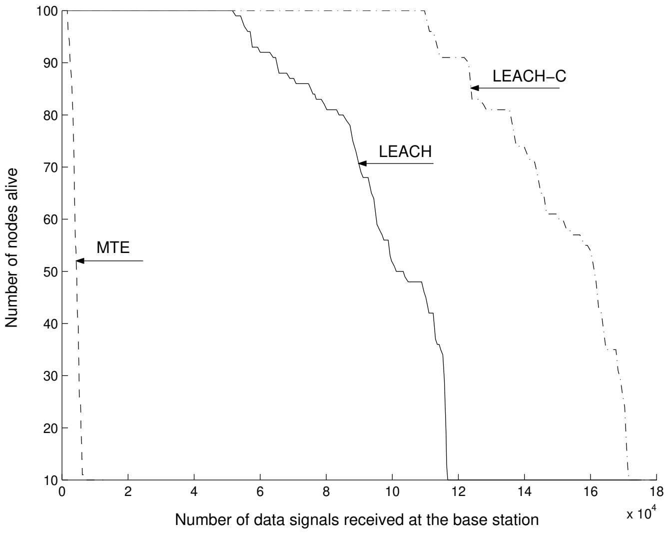


(c)

Figure 4-10: (Cont.) Data for the unequal energy simulations, where 10 nodes began with 200 J of energy and the remaining 90 nodes began the simulations with 2 J of energy. (c) The total amount of data received at the base station per given amount of energy. LEACH can deliver an order of magnitude more data per unit energy as MTE.



(a)



(b)

Figure 4-11: Data for the unequal energy simulations, where 10 nodes began with 200 J of energy and the remaining 90 nodes began the simulations with 2 J of energy. (a) Number of nodes alive over time. (b) Number of nodes alive per amount of data sent to the base station. Since MTE cannot take advantage of the high-energy nodes, it cannot send much data to the base station before nodes begin to die. LEACH can send over 50 times the amount of data for a given number of node deaths as MTE routing.

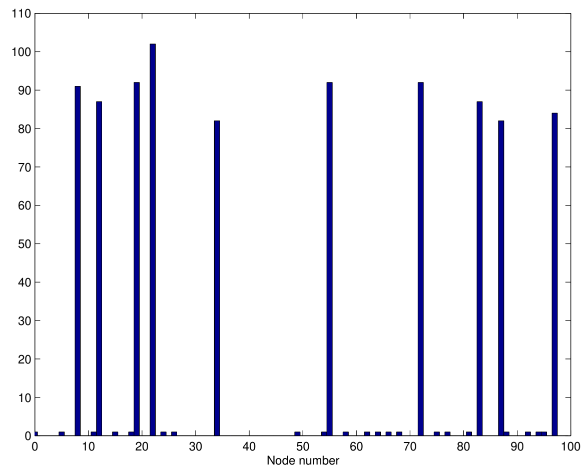


Figure 4-12: Total number of times each node was cluster-head in LEACH. The 10 nodes that began with 200 J were cluster-heads much more often than the 90 nodes that began with 2 J.

4.5.4 Cluster Formation Costs

While LEACH-C produces superior clusters to LEACH, there is a cost to using a centralized cluster formation algorithm compared to a distributed algorithm. This protocol requires a GPS or other location-tracking device on the nodes, and the start-up phase is more energy-intensive than the distributed approach since information from each node must be transmitted to the base station at the beginning of each round. The total energy dissipated in cluster formation was tracked for both LEACH and LEACH-C. For LEACH, this startup energy includes the energy for each of the cluster-head's advertisement messages, non-cluster-head nodes' join-request messages, and transmission/reception of the TDMA schedule in each cluster. For LEACH-C, the startup energy includes transmission of a small message containing node location and current energy from each node to the base station (using CSMA) and the reception of the cluster information from the base station (note that these results do not include any energy costs for the base station or for carrier-sense). The total energy dissipated during cluster formation is shown in Figure 4-13. The startup energy at the beginning of each round is constant in LEACH-C because the same number of transmissions and receptions occur in each round (using CSMA with a large amount of transmit power to reach the base station, the probability of collision is small). In LEACH, however, there is a different amount of energy for cluster formation at the beginning of each round, as the total energy will depend on the number of nodes that elect themselves to be cluster-heads and their locations within the network. A larger number of cluster-head nodes implies more energy dissipated in the

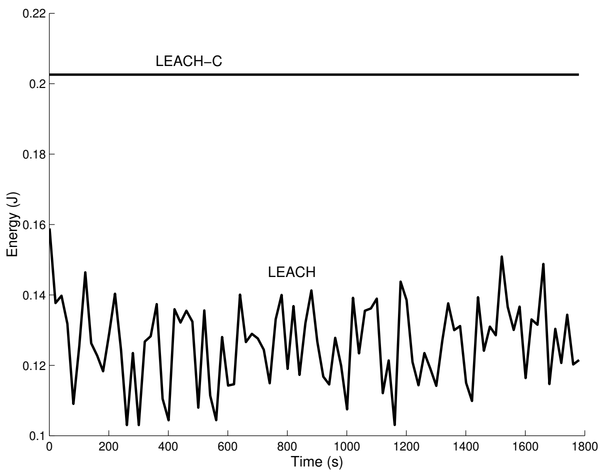


Figure 4-13: Amount of energy expended during cluster formation for the distributed algorithm (LEACH) and the centralized algorithm (LEACH-C).

network. This graph shows that LEACH-C cluster formation dissipates almost twice as much energy as LEACH cluster formation. However, despite this increase in startup energy dissipated, overall, LEACH-C is more energy-efficient than LEACH because the centralized algorithm can determine better clusters than the distributed algorithm.

4.6 Summary and Future Work

This chapter has shown the large advantage of using LEACH versus a routing protocol or static clustering approach in terms of latency and system lifetime for a given quality (measured here as the amount of effective data whose information is received at the base station). LEACH is effective because it was designed to exploit the application of sensor networks: producing high-level information about an environment the nodes are monitoring. LEACH-C is able to deliver more effective data than LEACH even though cluster formation is more expensive because the centralized algorithm can use network topology information to form good clusters that require less energy for operation than the ad-hoc clusters formed in LEACH. However, this protocol comes at the price of having to know node location. Similarly, LEACH-F performs well, but this protocol is unable to adapt to new conditions, such as nodes being added to the network or node mobility. Therefore, this protocol does not meet the robustness requirements for wireless microsensor networks.

In order to more accurately compare the different protocols, we need to create an event-driven simulator. Such a simulator would be able to measure the probability of missed detections and false alarms of events occurring in the environment. Thus, rather then using the amount of data (or effective data) received at the base station as a measure of the quality the protocols achieve, using an event-driven simulator would allow us to get a more realistic and application-specific determination of the quality of different network protocols. This is a subject for future work.

In our simulations, we modeled the energy dissipated when a node communicates data or performs computation on the data. In addition to the energy dissipated while the radio is transmitting and receiving or computing on data, a node will dissipate static energy. Nodes are often described as being in one of the following states: sleep, idle, or active. If the node is in the sleep state, the radio and processor are turned off and only the sensor module is active. In this state, the node dissipates P_{sleep} W. In the idle state, all modules are turned on and are ready to perform but not currently processing data or radio signals. The power dissipated in this state is P_{idle} W. Finally, the node is in the active mode if it is transmitting, receiving, or computing on data. The power dissipated in this state is equal to the idle power plus the power required to perform the function. Therefore,

$$\begin{aligned} P_{Rx-active} &= P_{idle} + P_{Rx} \\ &= P_{idle} + E_{Rx}(l)R_b \end{aligned} \tag{4.34}$$

$$\begin{aligned} P_{Tx-active} &= P_{idle} + P_{Tx} \\ &= P_{idle} + E_{Tx}(l,d)R_b \end{aligned} \tag{4.35}$$

$$\begin{aligned} P_{compute-active} &= P_{idle} + P_{compute} \\ &= P_{idle} + E_{compute}(l)R_b \end{aligned} \tag{4.36}$$

Using this static power dissipation model, protocols that allow nodes to remain in the sleep state as long as possible obtain a large advantage over protocols that do not (e.g., TDMA allows nodes to go to sleep whenever they are not transmitting data, whereas FDMA requires nodes be on at all times, transmitting at a lower rate). LEACH was designed to ensure nodes could go into the sleep state often and should thus perform well under this static power dissipation model.

In a large, distributed network, it may be advantageous to have the cluster-head nodes form a multi-hop backbone to get data from the cluster-head nodes to the base station. In this case, aggregate signals would be sent along a pre-determined route until they reached the base station. This may save energy by reducing the long-distance transmissions between far-away cluster-head nodes and the base station.

Finally, it would be beneficial to compare LEACH to some of the new "power-aware" routing protocols (e.g., [85, 70]). Our intuition is that power-aware routing will increase the system lifetime compared to MTE routing, since the energy dissipation is more evenly distributed among the nodes, but that the overall energy and latency efficiency will probably decrease slightly, since longer routes that avoid hot-spots will be chosen.

An interesting question arises from the results of our experiments: can data aggregation be used with a routing protocol to get better performance? This can be done, but it requires that data that comes from sensors possibly far from each other be aggregated together. On the other hand, using a clustering approach ensures that data that is aggregated comes from sensors located spatially close to each other. If we assume that all data received by the nodes is correlated and can be aggregated together, we can use a *wave*-type protocol, where data propagate closer to the base station and are aggregated along the way. This protocol works as follows:

- Each node keeps track of its upstream neighbors, i.e., the nodes that will be sending it their data.
- Each node that does not have any upstream neighbors uses a CSMA protocol to send data to its next-hop neighbor at the beginning of a round.
- Once a node has received data from every upstream neighbor, the node aggregates the data together (along with its own data) and forwards the aggregate signal to its next hop neighbor (again, using CSMA).
- This continues until the data works its way to the base station.

For example, Figure 4-14 shows an example of this protocol. The nodes at the edge of the network (e.g., those with no upstream neighbors) send their data first (Figure 4-14a). Once a node receives all the data from its upstream neighbors, it aggregates the data with its own and sends the aggregate signal to its next-hop neighbor (Figure 4-14b). This continues until all the data has been aggregated together and sent to the base station (Figure 4-14c). Using this protocol, each node sends its aggregated data signal a short distance (except the end nodes that must transmit their aggregated data a long distance to the base station). Therefore, this protocol has the advantage of being low-energy for most nodes. Since the data are being compressed within the network, this protocol is much more energy- and latency-efficient than MTE routing. However, the drawback of such an approach is that if the data are not correlated, the quality of the results produced by this approach may be degraded.

This chapter showed the advantage of using an application-specific protocol architecture for wireless microsensor networks. This approach can be used for conventional networks, such as multimedia networks, to achieve high performance from the network as well. This is discussed in the next chapter, where we describe our research creating an application-specific link-layer design that improves the reconstructed video quality for MPEG-4 compressed video transmitted over wireless networks.

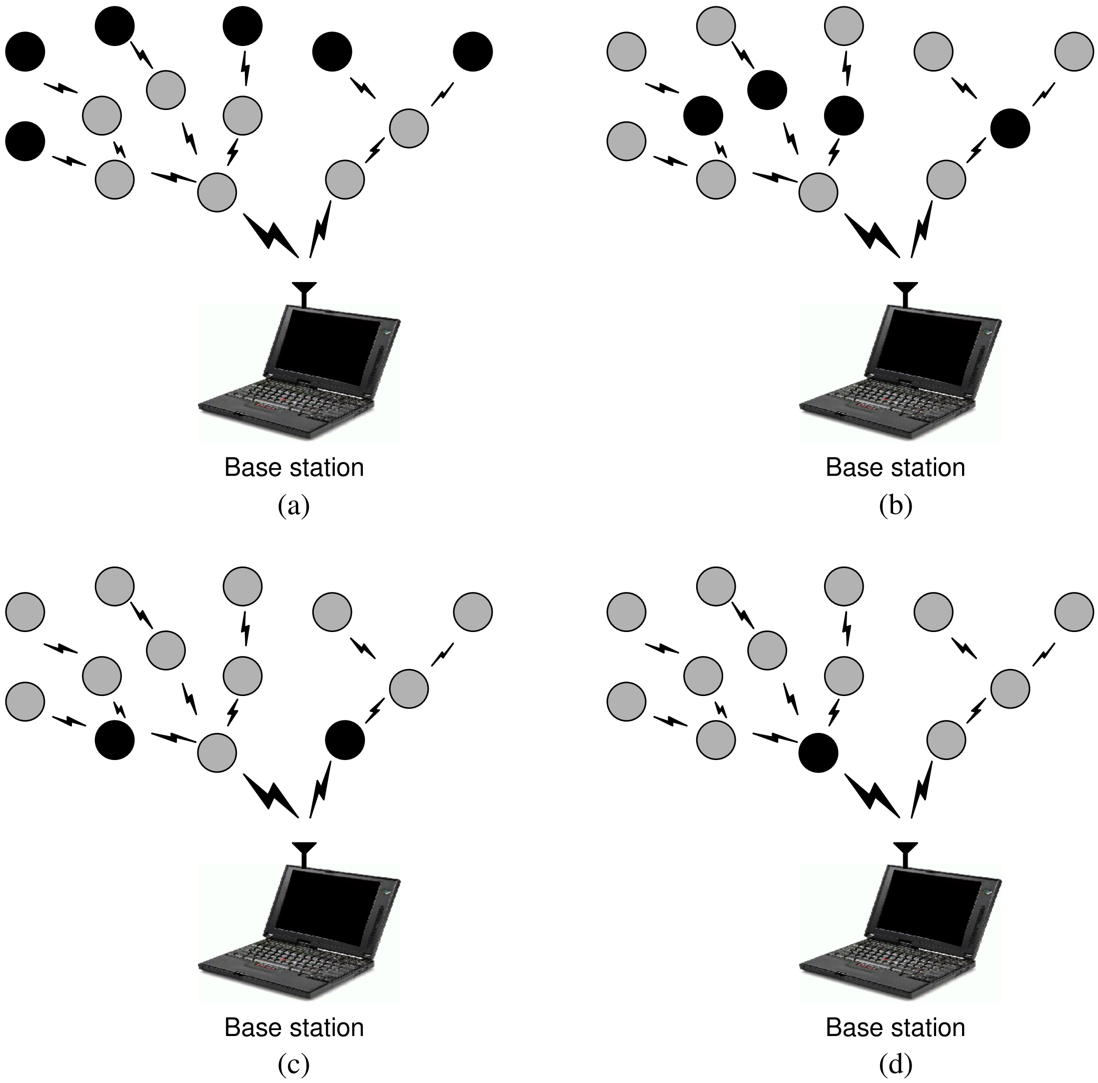


Figure 4-14: The "wave" protocol architecture. If data from all the nodes are correlated, data aggregation can be performed within a routing protocol architecture. In the wave protocol, nodes wait until they receive all the data from their upstream neighbors, then they aggregate the data with their own and send the aggregate signal to their next-hop neighbor.

Chapter 5

Error Protection for Wireless Video Systems

In the previous chapter of this dissertation, we showed the advantage of using a cross-layer protocol architecture for wireless microsensor networks. As the function of these networks is very different than that of traditional, data-oriented wireless networks, we would expect to be able to exploit this to create protocol architectures better suited for wireless microsensor networks than general-purpose protocol architectures. Can we achieve similar gains by designing application-specific protocols for traditional wireless networks? In these types of networks, the user creates the data, so there is no correlation among data from different users. In addition, it *is* important that the end-user obtain all data. However, the transmission requirements are very different depending on what type of data are being transmitted. For example, delay is unacceptable for real-time audio and video delivery, but a system that decreases energy dissipation by increasing latency is advantageous for data delivery. As the constraints on the portable device due to energy limitations and the nature of the wireless channel are stringent, we would like to obtain the best possible performance for the least cost.

When sending video over wireless networks, the cost is the energy required to send the information and the latency in data transfer. To reduce these costs, the number of bits that need to be transmitted should be reduced. The performance of a wireless video system is the quality of the reconstructed video. There is an inherent trade-off between cost and performance: the more bits of information that can be sent, the higher reconstructed video quality can be obtained. Therefore, the goal in designing a wireless video system is to maximize reconstructed video quality for a given number of bits transmitted over the channel (or, alternatively, to minimize the number of bits transmitted over the channel to obtain a certain reconstructed video quality). One way to accomplish this is to use cross-layer design of the link-layer (channel coding) protocol. A system that exploits application-specific information about the bitstream will produce higher quality for the same number of transmitted bits [39, 102].

5.1 Multimedia Standards

Researchers have been developing sophisticated compression and communication algorithms for the past couple decades. The compression algorithms aim to reduce the amount of information needed to represent the original data (usually trading off the quality of the representation for the amount of data reduction) while the communication algorithms enable the compressed data to be successfully received after transmission by adding appropriate packet headers, multiplexing (or interleaving), and channel coding. In order to facilitate interoperability, standard compression and communication algorithms have been created. Standards-compliant devices can communicate with each other, even if they were created by different manufacturers. Therefore, it is advantageous to fit new protocols within existing standards. We shall show that our unequal error protection link-layer protocol, which exploits information about an MPEG-4 compressed video bitstream, can be fit into the H.223 communication protocol.

5.1.1 H.223 Communication Standard

As part of the H.223 multiplex standard [34, 35], an adaptation layer may be used to provide additional protection from channel errors, beyond the level of service available from the network provider. The adaptation layer of the H.223 standard provides support for forward error correction (FEC) using rate-compatible punctured convolutional (RCPC) encoding of the data (see Section 2.1.1 for details about FEC and RCPC coding). The amount of protection can be set based on the channel conditions and the amount of allowed overhead to bring the aggregate bit error rate down to a level that is acceptable to the application.

In a typical use of H.223, the application passes separate audio, video, and data streams to the H.223 adaptation layer. Each of these streams is independently FEC-coded, and the coded data are sent to the multiplex layer. The multiplex layer performs multiplexing of the streams and adds a resynchronization flag and a header to the multiplexed data (the payload). This flag is chosen

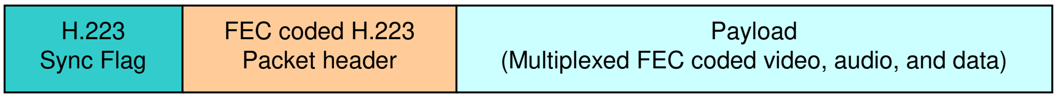


Figure 5-1: H.223, a multiplexing protocol for low bitrate multimedia communication, supports channel coding in the adaptation layer, before video, audio, and data streams are multiplexed to form the payload of an H.223 packet. A synchronization flag and packet header further protect the packet against channel errors.

so that it has good auto-correlation properties and has low cross-correlation with the data in the payload. Detection of the resynchronization flag is done at the H.223 decoder using correlation and thresholding. This allows a high degree of detection and a low degree of false detection in the presence of channel errors. The header added by the H.223 multiplex layer contains the length of the payload and a code into a multiplex table that tells the decoder how to demultiplex the video, audio, and data. This header is protected using an extended Golay error correction code. Figure 5-1 shows the structure of an H.223 packet.

The H.223 packets are sent over a bandwidth-constrained, error-prone wireless channel. At the receiver, the (possibly corrupted) packets are demultiplexed and FEC decoded using the multiplex and adaptation layers of H.223, respectively.

5.1.2 MPEG-4 Simple Profile Video Compression

Mobile multimedia terminals must be able to compress video for transmission over low-bandwidth, error-prone wireless networks such that the decoder obtains high quality reconstructed video. The latest MPEG standard, MPEG-4, uses a hybrid block motion compensation/discrete cosine transform (BMC/DCT) coding technique to achieve large amounts of compression (see Figure 5-2). Frames are either coded in intra mode (I-frames) or inter mode (predicted frames, or P-frames). To code an I-frame, the frame is broken into 16 pixel $\times$ 16 pixel blocks called macroblocks. Each macroblock is further broken into four 8 pixel $\times$ 8 pixel blocks that are transformed using the DCT. The DCT coefficients are quantized and run-length encoded, and the (run,length) pairs are further compressed using variable-length codewords. P-frames are predicted from the previous frame using BMC. Using motion estimation, each macroblock (or each 8 pixel $\times$ 8 pixel block) is matched with the closest version of the block with the pixels in the previous frame. A motion vector is generated

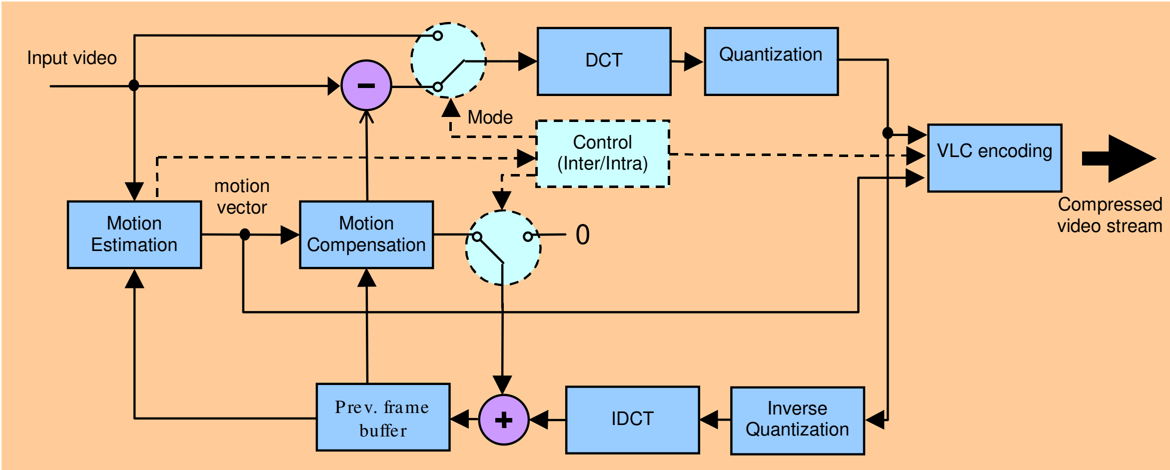


Figure 5-2: Block diagram of an MPEG-4 video coder. MPEG-4 uses block motion compensation (BMC) and the discrete cosine transform (DCT) for compression. The DCT coefficients are quantized, run-length encoded, and variable-length coded.

that points to the best match location[1]. The difference between the best prediction and the actual macroblock is broken into 8 pixel × 8 pixel blocks and transformed using the DCT. As with I-frames, the DCT coefficients are quantized, run-length encoded, and the (run,length) pairs are compressed using variable-length codewords. Unlike previous video compression standards, MPEG-4 does not have a required pattern of I- and P-frame coding. MPEG-4 instead supports the use of intra-coded macroblocks within a P-frame (called intra-refresh). This means that certain blocks within a P-frame can be compressed without using BMC. The encoder can decide when to intra-refresh each macroblock (representing a tradeoff between compression and quality, as intra-coded macroblocks require many more bits to code than inter-coded macroblocks but produce higher quality), but each macroblock must periodically be intra-refreshed to avoid numerical error accumulations [16].

The use of predictive coding of the frames and variable length codewords in MPEG-4 makes the compressed video bitstream sensitive to channel errors, as predictive coding causes errors in the reconstructed video to propagate in time to future frames of video, and the variable-length codewords cause the decoder to easily lose synchronization with the encoder in the presence of bit errors. To make the compressed bitstream more robust to channel errors, the MPEG-4 video compression standard incorporated several error resilience tools to enable detection, containment,

[1]There will either be 1 motion vector per macroblock or 4 motion vectors per macroblock, determined by the encoder.

and concealment of errors. These tools include resynchronization markers, header extension codes, data partitioning, and reversible variable-length coding [89, 90].

Resynchronization Markers

When using the resynchronization marker option in MPEG-4 compression, the video bitstream is broken into *video packets* that contain data for an integer number of macroblocks. Each video packet begins with a resynchronization marker, a unique 17-bit code that cannot be emulated by any combination of the codewords in the compressed video. This marker consists of 16 zeros followed by a single 1. Following the resynchronization marker is header information that is needed to decode the macroblock data contained within the packet. This header information includes the macroblock number of the first macroblock whose data is included in the packet and the initial quantization step-size for the macroblock data in the packet. Once the decoder knows this header information, it can correctly decode and display the data from the video packet, even if previous packets were received in error.

MPEG-4 allows several means for breaking the bitstream into video packets. One recommended method is to keep the size of video packets approximately constant. The encoder determines the minimum number of bits n for the video packet. Once the encoder has sent n bits to the buffer, it finishes compressing the data of the macroblock it is currently coding, and this ends the video packet. This means that the number of macroblocks within each video packet is not constant. In particular, areas that contain a large amount of activity and hence require a large number of bits to encode, will have data in more video packets than areas with relatively little activity. This is advantageous because if there is an error in a packet that contains macroblocks with a large amount of motion from the previous frame, the decoder should be able to resynchronize with the encoder as quickly as possible to prevent the loss of large amounts of data that are visually important.

Header Extension Codes (HEC)

This is a 1-bit code that, when set to 1, signifies that important header information follows. This header information includes frame parameters such as the spatial dimensions of the frame, the type of frame (intra or inter), and the temporal reference of the frame relative to the previous frames. This information is typically included only once, at the beginning of the frame. If this information is corrupted, the decoder must discard all data corresponding to the frame. To reduce the sensitivity to errors at the beginning of the frame, the encoder can additionally include this frame header information in the header of a number of video packets. If the HEC bit is set to 1, the decoder knows that additional frame header information follows. If it is set to 0, no such information is present in this packet. Using HEC results in fewer discarded frames than when it is not used.

Data Partitioning

If an error occurs in a video packet, typically all data corresponding to macroblocks in that packet must be discarded, since it is not known exactly where the error occurred. Concealment is then typically performed by copying the macroblocks from the same location in the previous frame. If there is any motion between frames, this will result in poor reconstructed quality. In order to enhance the quality, MPEG-4 includes a mode called data partitioning. If this mode is used, the macroblock data within a packet are broken into motion information and texture (DCT) data, with a unique motion-marker placed between these two segments. The advantage of using data partitioning is that if an error occurs in the texture data, only the texture data of the macroblocks in the packet must be discarded. If the motion information for the macroblocks is correctly decoded, it can be used to perform motion-compensated concealment. In this case, rather than copying the block in the same spatial location from the previous frame, the decoder can copy the motion-compensated block specified by the motion-vectors of each macroblock whose texture data was corrupted. Motion-compensated concealment results in much higher quality video.

Reversible Variable-Length Codes

The DCT coefficients in transformed video are quantized and then run-length encoded using variable-length codewords. MPEG-4 includes the option of using reversible variable-length codewords. These are codewords that can be decoded in the forward or reverse direction. Therefore, if the decoder detects an error in the bitstream and loses synchronization with the encoder, it can skip to the next resynchronization marker and decode backwards to obtain more data. Using reversible variable-length codes enables more DCT data to be recovered in the event of an error.

The output of an MPEG-4 simple video encoder that uses all 4 error resilience tools is a bitstream that contains video packets that begin with a header, which is followed by the motion information, the texture information (DCT coefficients), and stuffing bits (see Figure 5-3). The header bits represent the most important information of the packet, since the whole packet will be dropped if the header is received in error. The motion information has the next level of importance, as motion-compensation cannot be performed without it. The texture information is the least important of

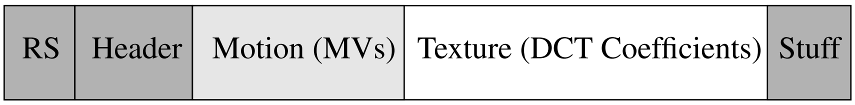


Figure 5-3: Video packet from an MPEG-4 encoder. If resynchronization is used, the packet begins with a unique resynchronization marker (RS) that cannot be emulated by the bits in the video packet. Following the resynchronization marker is a header that contains all the information needed to decode the data in the video packet. If data partitioning is used, the data are broken into motion and texture information. All the data corresponding to motion information are placed at the beginning of the payload, whereas all the data corresponding to texture information are placed at the end. The video packet ends with stuffing bits that ensure the resynchronization marker is byte-aligned.

the four segments of the video packet. Without the texture information, motion-compensated concealment can be performed without too much degradation of the reconstructed picture. The stuffing information at the end of the packet has the same priority as the header bits because reversible decoding cannot be performed if this information is corrupted and the following packet may be dropped if the stuffing bits are received in error. The structure of the MPEG-4 video packet, where some bits are more important than other bits, can be exploited in the data-link processing functions to achieve higher application-perceived quality than can be achieved using a general-purpose data-link protocol.

5.2 Unequal Error Protection of MPEG-4 Compressed Video

The MPEG-4 error resilience tools are effective against a certain level of bit errors. Through experiments, we found that as long as the channel bit error rate is less than approximately 10^{-3}, the decoder can produce acceptable quality reconstructed video. However, wireless channels can have much higher bit error rates (e.g., carrier-to-interference ratios (C/I) as low as 4 dB $\approx$ 10% BER [92]). Therefore, channel coding is needed to bring the effective bit error rate in the source-coded bitstream down to an acceptable level.

The video packets from an MPEG-4 encoder have explicit structure that can be exploited by adding an unequal amount of error protection to the compressed bitstream. The goal of video compression algorithms is to spend more bits representing visually relevant data at the expense of fewer bits representing the less visually important data to try to minimize the visual distortion of the compression for a fixed rate. This idea can be extended to channel coding of MPEG-4 video packets. Since the packets can be broken into sections with different levels of importance to the

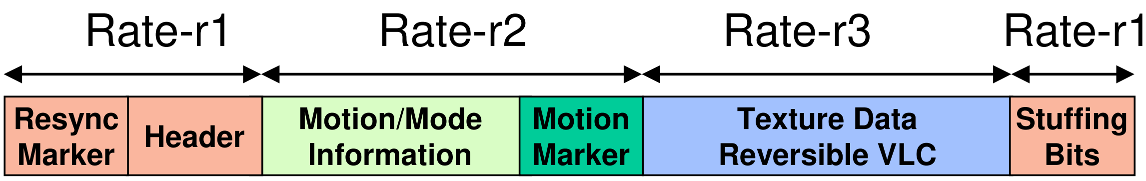


Figure 5-4: Unequal protection of an MPEG-4 video packet. To ensure the more important bits are protected the most and the least important bits are protected the least, $r_1 < r_2 < r_3$.

application, the number of bits for channel coding each section should be proportional to its relative importance, creating an *unequal error protection (UEP)* channel coder[2].

5.2.1 System Overview

Using unequal error protection implies that different rate coders are applied to different sections of the video packet. When using unequal error protection, the header and stuffing bits would get the highest amount of protection since they are the most important bits of the video packet, the motion bits would get the next highest level of protection, and the texture bits would receive the lowest level of protection. Figure 5-4 shows the channel coders that are applied to each of the different sections of the video packet. To match the coders to the level of importance of the different sections of the packet, the coder rates are chosen such that $r_1 < r_2 < r_3$. Using this system, the errors are less likely to occur in the important sections of the video packet, thereby improving application-perceived quality.

If the number of bits in the MPEG-4 video packet is:

$$X_{Total} = X_{Header} + X_{Motion} + X_{Texture} + X_{Stuffing} \tag{5.1}$$

then the number of bits in the FEC-coded video packet is:

$$Y_{UEP} = \frac{X_{Header}}{r_1} + \frac{X_{Motion}}{r_2} + \frac{X_{Texture}}{r_3} + \frac{X_{Stuffing}}{r_1} \tag{5.2}$$

For equal error protection (EEP), $r_1 = r_2 = r_3 = r$ and

$$Y_{EEP} = \frac{X_{Total}}{r} \tag{5.3}$$

[2]The work described in this section was performed at Texas Instruments.

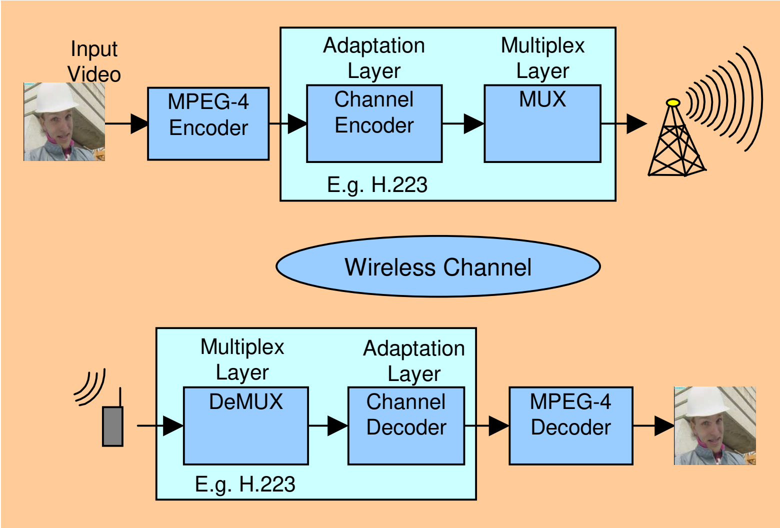


Figure 5-5: System-level view of an H.223 coder used with an MPEG-4 source coder in a wireless environment. Unequal error protection can be incorporated into the adaptation layer of the H.223 coder.

If there is a fixed overhead, the UEP channel coder rates must be chosen to ensure that $Y_{UEP} = Y_{EEP}$.

After source and channel coding, the FEC-coded bitstream must be packetized for transmission across the wireless channel. This can be accomplished using the H.223 multiplexing protocol. In a typical H.223 system, the adaptation layer adds a fixed amount of FEC to the audio, video, and data bitstreams, and a multiplex layer then multiplexes the different bitstreams in a pre-specified manner. The standard H.223 adaptation layer can be replaced with the channel coder described above to enable unequal error protection. The FEC-coded bitstream is sent from the adaptation layer/channel coder to the multiplex layer, where the video is multiplexed with other bitstreams, and a header and resynchronization marker are added. These H.223 packets are sent over a wireless channel, where they get corrupted. The packets received at the decoder are de-multiplexed, channel decoded and source decoded to obtain the reconstructed video. Figure 5-5 shows the entire video transmission system.

5.2.2 Experimental Set-up

To test the use of unequal error protection, we ran several experiments using the sequences "Akiyo" and "Mother & Daughter" at both common intermediate format (CIF, 496×384 pixels) and quarter-CIF (QCIF, 248×192 pixels) resolution. The quantization parameter was chosen so that the source coding output was approximately 48 Kbps at 7.5 fps for the CIF images and 24 Kbps at 10 fps for the QCIF images. Each reconstructed sequence contained 10 seconds of video.

The sequences were coded using all the MPEG-4 error resilience tools. The compressed bitstreams were then channel coded using convolutional encoding of the data with either equal error protection using a fixed rate-$\frac{7}{10}$ code or unequal error protection using a rate-$\frac{3}{5}$ code for the header and stuffing segments, a rate-$\frac{2}{3}$ code for the motion segment, and a rate-$\frac{3}{4}$ code for the texture segment. These EEP and UEP rates, chosen because they ensure that $Y_{UEP} \approx Y_{EEP}$, were obtained by puncturing the output of a rate-$\frac{1}{2}$ code that was produced by the two polynomials [9]:

$$g_1(X) = X^6 + X^5 + X^3 + X^2 + 1 \tag{5.4}$$

$$g_2(X) = X^6 + X^3 + X^2 + X + 1 \tag{5.5}$$

The FEC-coded sequences were sent through a MUX, and the output packets from the MUX were sent through a GSM channel simulator. This simulator is based on a complex model of a GSM channel that has been fitted with data taken from a real GSM channel to get an accurate account of the errors found on this channel. The channel is not a binary channel, so bits are sent with a given "power" level. The received power is attenuated from the effects of transmission through the channel.

Each FEC coded bitstream was subjected to 6 different GSM channel conditions ranging from 0.3% to 12% BER (corresponding to a carrier-to-interference ratio of between 19 dB and 4 dB) in 50 different trials per channel condition. For each of these trials, the first frame was transmitted without corruption. The corrupted bitstreams were channel decoded and the error-corrected bitstreams were source decoded to find the quality (average PSNR) of the reconstructed video.

5.2.3 Results

In order to compare the different methods of adding channel coding to the compressed video, the results from the 50 trials at a given GSM channel error rate were averaged for both sequences.

Figure 5-6 shows the average BER that remains after channel decoding for each of the GSM channel BER conditions. Channel coding reduces the effective BER seen by the video decoder by over an order of magnitude for most of the raw channel conditions. However, the convolutional codes break down when the channel error rate is too high. Thus for the GSM channels with a BER around 10%, the channel coding actually increases the effective BER seen by the decoder.

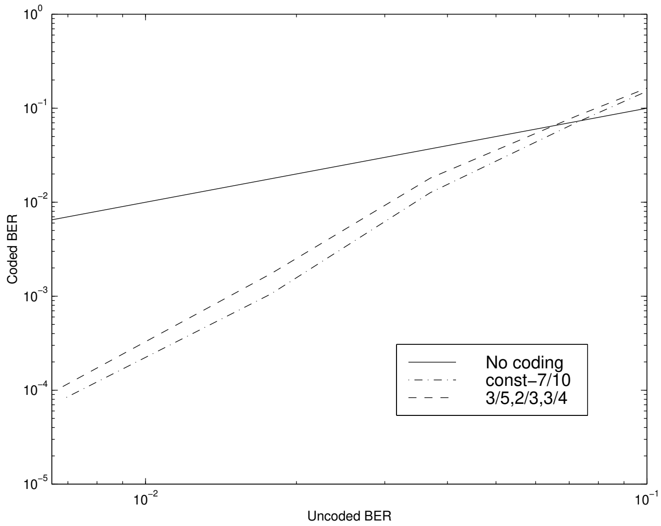


Figure 5-6: Effective BER for EEP and UEP. Channel coding reduces the effective BER seen by the video decoder by over an order of magnitude for most of the raw channel conditions.

Under such harsh conditions, the channel coder would need to use more powerful codes to reduce the BER. However, for the remainder of the GSM channel conditions, the FEC codes reduce the effective BER. This brings the number of bit errors remaining in the bitstream that is sent to the MPEG-4 decoder to a level at which the error resilience tools can work.

Figure 5-7 shows a comparison of the average PSNR values obtained for fixed coding and unequal error protection. These plots show that unequal error protection produces the highest average PSNR for the reconstructed video for both CIF and QCIF images at high channel error rates. Since both coding methods require the same amount of FEC overhead, this improvement (as much as 1 dB) does not require additional bandwidth. In addition, for the error conditions shown here, the fixed rate-$\frac{7}{10}$ coder actually produces fewer errors in the channel decoded bitstream than the UEP coder (as shown in Figure 5-6), yet it still produces lower quality reconstructed video. This is because the errors are spread evenly throughout the different portions of the video packet. Conversely, the unequal error protection coder may leave more errors in the channel decoded bitstream, but these errors are in less important portions of the video packet.

Figure 5-8 shows a reconstructed frame of "Akiyo" when there are no channel errors and when the GSM channel error rate is 4% and the video is protected using EEP with a rate-$\frac{7}{10}$ coder and UEP with a rate-($\frac{3}{5}$, $\frac{2}{3}$, $\frac{3}{4}$) coder. These images also show the advantage of using unequal error protection.

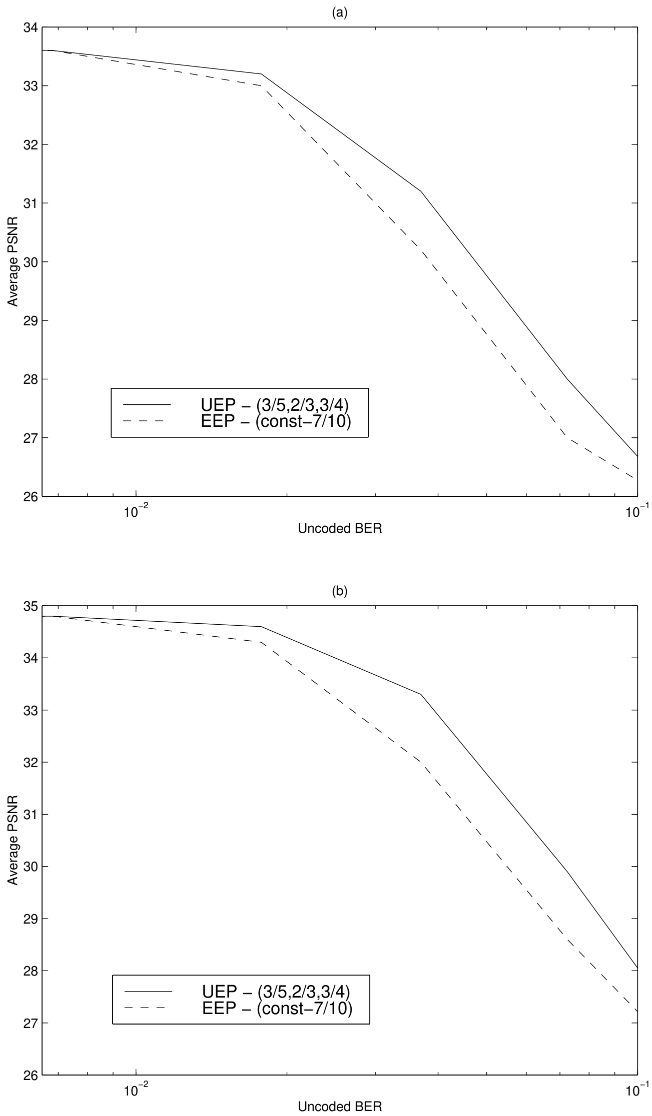


Figure 5-7: Average PSNR for EEP and UEP channel coding of MPEG-4 video compressed with all the MPEG-4 error resilience tools. (a) CIF images. (b) QCIF images. UEP produces higher PSNR than EEP even though there are more errors in the bitstream sent to the MPEG-4 decoder (as seen in Figure 5-6). This is because these errors are in less important sections of the video packet. Therefore, UEP achieves higher *application-perceived* quality than EEP.

(a)

(b)

(c)

Figure 5-8: Comparison of a frame of "Akiyo". (a) shows the reconstructed frame with no channel errors, and (b) and (c) show the reconstructed frame after transmission through a simulated GSM channel with 4% BER using (b) EEP coding and (c) UEP coding. UEP produces visibly better images than EEP for the same amount of channel-coding overhead.

Rather than using the extra bandwidth for channel coding, it might be beneficial to spend these bits on forced intra-MB updates. These intra-MBs would stop error propagation and hence improve reconstructed video quality. In order to test the effectiveness of using intra-MBs, the video sequences were compressed with enough forced intra-MBs each frame to increase the source-coded bitrate to equal that of the FEC-coded bitstream when no intra-MBs are used. The results of this experiment are shown in Figure 5-9, labeled “No coding (Intra refresh only)”. These plots show that it is much better to use the overhead for channel coding than forced intra-MBs at these high channel error rates. Using the overhead for intra-MB refresh increases the number of source bits that are corrupted due to channel errors, causing the reconstructed quality to be poor. As the channel error rates decrease below the levels tested here, it would probably be advantageous to reduce the number of bits spent on channel coding and increase the number of forced intra-MBs per frame to get the optimal reconstructed video quality.

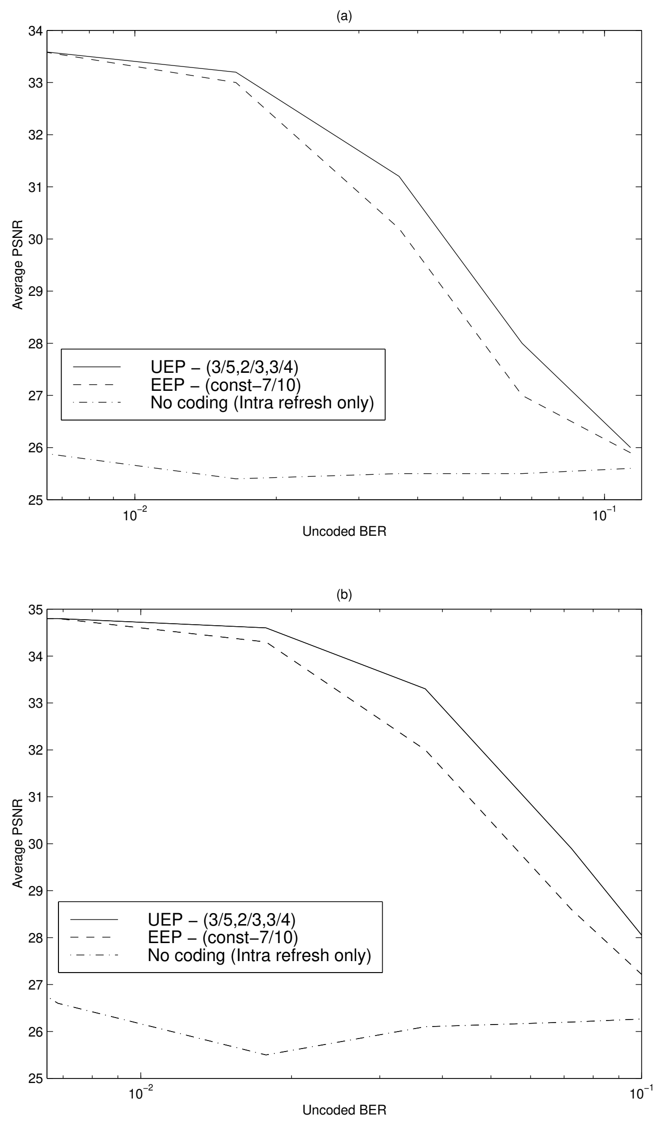


Figure 5-9: Average PSNR for EEP and UEP channel coding of MPEG-4 video compressed with the all the MPEG-4 error resilience tools. These graphs also show the average PSNR when no channel coding is added to MPEG-4 video that is coded using extra intra-MBs to give the source-coded bitstream the same number of bits as the channel-coded bitstreams. (a) CIF images. (b) QCIF images. At these high channel BERs, it is better to use the overhead for channel coding than to add intra-MBs. However, as the channel error rate decreases, it would probably be advantageous to use fewer overhead bits for channel coding and more for intra-MBs.

Chapter 6

Conclusions

Use of the wireless channel is growing at an amazing speed. Advances in energy-efficient design have created new portable devices that enable exciting applications for the wireless channel. While the wireless channel enables mobility, it adds constraints that are not found in a wired environment. Specifically, the wireless channel is bandwidth-limited, and the portable devices that use the wireless channel are typically battery-operated and hence energy-constrained. In addition, the wireless channel is error-prone and time-varying. Therefore, it is important to design protocols and algorithms for wireless networks to be bandwidth- and energy-efficient as well as robust to channel errors. This can be accomplished using cross-layer protocol architectures, that exploit application-specific information to achieve orders of magnitude improvement in bandwidth and energy efficiency and improvements in application-perceived quality. The work described in this dissertation has demonstrated the advantages of application-specific protocol architectures by designing and evaluating protocol architectures for two different application spaces: large-scale, distributed microsensor networks and wireless transport of compressed video.

6.1 Summary of Contributions

When designing cross-layer protocol architectures for wireless networks, it is important to clearly define the goals and requirements of the system. This will enable the designer to make good trade-offs in the different system parameters to best support the goal of the application. We have analyzed two different types of networks to determine the important system parameters and application requirements: wireless microsensor networks and wireless video transport networks. Based on the design constraints, we developed application-specific protocol architectures that provide large benefits to the application.

For wireless microsensor networks, we designed LEACH with the following features to meet the design criterion we identified:

- Ease of deployment. In order to ensure that the nodes can be easily deployed in remote, hostile, or difficult areas, LEACH is self-configuring. The set-up protocol for LEACH uses a distributed algorithm whereby nodes make autonomous decisions that result in all nodes being placed into clusters. In addition, LEACH uses the received signal strength of communication messages to determine "communication distance" between nodes, rather than relying on distance information. This ensures that nodes do not need to be placed in specific, known locations.

- Maximum system lifetime. As the nodes are battery-operated, reducing energy dissipation helps extend system lifetime. LEACH uses low-energy MAC and routing to reduce energy-dissipation. These protocols were chosen to allow the nodes to remain in the sleep state, with some internal modules powered-down, for a large amount of operation time. Since we assume that all nodes have data to send, using a TDMA protocol, where each node is given a specific time-slot in which to transmit, achieves this goal. In this case, nodes only need to be awake during their specific slot to transmit data, and they never need to go into the idle mode, where they are waiting to see if any nodes have data to transmit.

  LEACH also minimizes energy dissipation by exploiting the data-gathering aspect of microsensor networks. Since LEACH is a cluster-based protocol, nodes within a cluster are located close to each other and thus are likely to have correlated data. Performing local data aggregation on the correlated data can greatly reduce energy dissipation when the energy required for computation is less than the energy required for communication.

  In addition, ensuring protocol robustness helps maximize system lifetime. If the protocol architecture was designed such that nodes depended on specific nodes to be functional, the loss of these nodes would have catastrophic affects on the network. Creating robust protocols ensures that this will not happen. In LEACH, the use of rotating cluster-heads and adaptive clusters ensures that node failures affects other nodes for a maximum of one round.

- Minimum latency. Reducing the amount of data that are transmitted throughout the network using application-specific data aggregation reduces the latency of getting the result to the end-user.

- Maximum quality. The notion of quality in a sensor network is very different than in a data or multimedia wireless network. In a sensor network, the quality refers to the high-level description of "events" in the environment, rather than individual nodes' data. Therefore, performing local signal processing to distribute the determination of events may have little impact on quality.

For wireless video systems, the number of bits that need to be transmitted (corresponding to energy dissipation and latency) and the reconstructed video quality are the important considerations. Protocol architectures for wireless video systems should be designed to obtain maximum quality for a fixed bitrate. We showed that this can be accomplished by adjusting the level of error-protection to the relative importance of the bits being protected. This ensures that the rate for channel-coding is matched to the level of distortion that would be created in the reconstructed video in the event of channel errors.

Although these ideas were developed for wireless microsensor networks and video transport networks, they can be applied to protocol architectures for general wireless networks. For example, ad-hoc wireless networks can benefit from the self-configuring protocol we developed for LEACH, and data-gathering networks can benefit from local data aggregation. Wireless data and multimedia networks can benefit from designing MAC and routing such that the nodes can go into the sleep state as long as possible, thereby saving considerable energy. Most data streams have some structure that can be exploited when designing link-layer protocols.

The research described in this dissertation contributes to our understanding of the benefits of designing protocol architectures using a cross-layer approach. We have developed and implemented application-specific protocol architectures for wireless microsensor networks and wireless video delivery systems that are better able to support the application than general-purpose approaches. All applications can benefit from protocol architectures that optimize for the their most important parameters by exploiting application-specific information.

6.2 Future Work

There is still much work to be done in the area of protocols for wireless microsensor networks. The protocols developed in this research are for scenarios where the sensors have correlated data. However, there are important applications of microsensor networks where this is not the case. For example, sensor networks for medical monitoring applications may have different sensors located on and/or in the body to monitor vital signs. These networks will not be as large-scale as the ones we discussed, but they will have similar requirements to the sensor networks we discussed—long system lifetime, low-latency data transfers, and high quality data. These networks will most likely focus on maximizing quality above all parameters, and loss of information will not be acceptable. Therefore protocol architectures need to be developed to support the unique considerations of these networks.

While we discussed the use of a beamforming algorithm to aggregate data, there is a need for better, faster, and more accurate data aggregation and classification algorithms. In addition to combining data from similar sensors, these algorithms need to handle multi-sensor input. For example, future sensor networks may contain cameras, microphones, and seismic sensors, each obtaining data about events in the environment. It is important to be able to aggregate this data, either from a high-level perspective (e.g., combining classifications from different sensors) or from the raw data (e.g., using the image and acoustic data together to classify the event).

Obtaining a better understanding of application-perceived quality will enable new protocols to make intelligent trade-offs between energy and quality. This will be particularly useful when energy is highly-varying, such as in self-powered systems (e.g., systems that convert vibration into electrical energy [63]). Protocols for self-powered systems will need to be adaptive to the current level of energy available and should produce the highest quality possible for a given energy to provide maximum benefit to the end-user. Such energy-quality scalability constraints add new parameters to the design of protocol architectures [10, 42].

Finally, it will be important to develop secure communication for wireless microsensor networks. End-users need to be able to ensure unauthorized users cannot access the data from the sensor networks. Furthermore, end-users need to be able to authenticate the data. Application-specific and scalable solutions may be able to provide the level of security required without draining the node's limited energy. Without these security measures in place, the application of sensor networks will be limited.

There is also a large amount of work that can be done to improve protocols for sending compressed data over wireless networks. While Shannon's source-channel separation theorem says the best performance possible is achieved when source and channel coding are performed separately, this only holds for infinite delay and complexity. As delay and complexity are limited for most applications, performing joint source-channel coding can improve performance over performing these functions separately. In our work, we implemented *source-assisted* channel coding, where information about the source coding was used to improve the channel coding. While our work showed advantages over approaches that do not incorporate this information into the channel coding, this work can be extended. Further research needs to be done to determine how to trade-off the number of bits allocated to source coding and the number allocated to channel coding. This will most likely depend on the channel state. For example, when the channel is bad, it is much more beneficial to the application to use fewer bits in source coding and add a great deal of controlled-redundancy to the data in the form of channel coding. As the channel improves, the large amount of channel coding is not necessary, and more bits should be spent on source coding to obtain the highest possible application-perceived quality. Determining how this trade-off should be done is the subject of future research.

In the future, home networking will enable all the devices in the home to communicate with each other. The new Bluetooth and HomeRF standards have created protocol architectures to support a range of data, from real-time voice and streaming video to asynchronous data transfers, for this scenario. Implementing adaptive protocols that change based on the application they are supporting will enable the network to produce high application-perceived quality for a range of devices connected to the same network. As in active networks, the packets themselves could encode information about how to decode/receive the packets sent from different applications.

As exemplified by our research, cross-layer design will enable wireless networks of the future to support the services required by different applications, helping us get closer to the goal of "anytime, anywhere" communication among and between users and devices.

Appendix A

ns Extensions

To implement LEACH and the general-purpose comparison protocols, we added several features to `ns` [67], an event-driven network simulator with extensive support for simulation of wireless network protocols. Developed at the University of California at Berkeley and the Lawrence-Berkeley National Laboratories in collaboration with the VINT (Virtual InterNetwork Testbed) project, `ns` has a simulation engine written in C++ with a command and configuration interface using OTcl. Network topologies can be easily described using the primitives Nodes, Links, Agents, and Applications, where Nodes represent end-hosts in the network, Links are the connectors through which Nodes communicate, Agents are used to implement different network protocols and are the points where packets are created and consumed, and Applications are used to generate data and perform different application-specific functions. Once the topology has been created, simulations can be run by starting the Applications on different nodes at various points in time.

While `ns` was developed as a simulator for wired networks, researchers at Carnegie Mellon University added extensive support for wireless networks. The CMU additions include mobile nodes, MAC protocols, and channel propagation models (that were described in Section 4.1.1. Figure A-1 shows the implementation of a mobile node. The Application class is written using the Tcl front-end, while the other functions that make up the node are written using the C++ engine. The Application creates "data packets" that are sent to the Agent[1]. The Agent performs the transport- and network-layer functions of the protocol stack. The Agent sends packets of data to CMUTrace, which writes statistics about the packets to trace files. The packets are then sent to a Connector which passes them to the Link-Layer for data-link processing. After a small delay, the

[1] In `ns`, data are not actually transferred to the nodes in the network. Rather, virtual data, in the form of a packet of a given length, is transmitted. Therefore, packets have a certain size but do not actually contain any data.

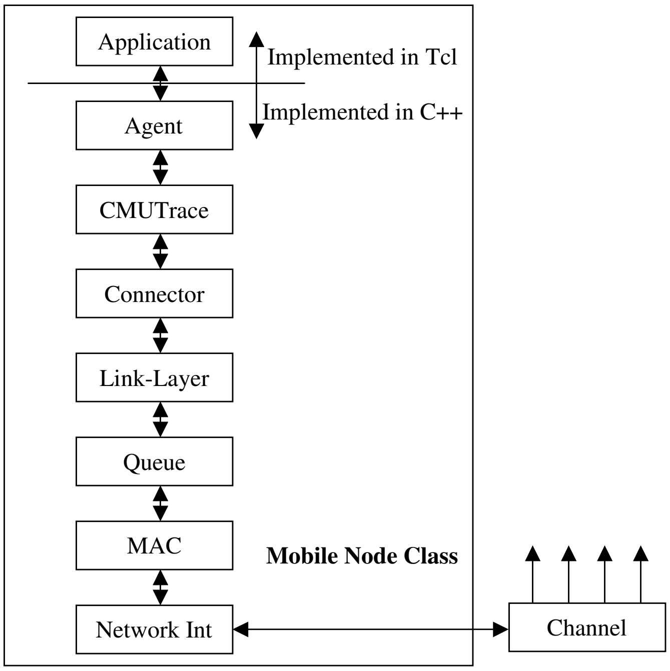


Figure A-1: Block diagram of an `ns` Mobile Node.

packets are sent from the Link-Layer to the Queue, where they are queued if there are packets ahead that have not yet been transmitted. Once a packet is removed from the Queue, it is sent to the MAC, where media access protocols are run. Finally, the packet is sent to the Network Interface, where the correct transmit power is added to the packet and it is sent through the Channel. The Channel sends a copy of the packet to each node connected to the channel. The packets are received by each node's Network Interface and then passed up through the MAC, Link-Layer, Connector, CMUTrace, and Agent functions. The Agent de-packetizes the data and sends notification of packet arrival to the Application.

A.1 Resource-Adaptive Node

We added a Resource-Adaptive Node [41] to `ns`, as shown in Figure A-2. The new features of a Resource-Adaptive Node are the Resources and the Resource Manager. The Resource Manager provides a common interface between the application and the individual resources. The Resources can be anything that needs to be monitored, such as energy and node neighbors. The Application can update the status of the node's resources through the Resource Manager using the functions:

- *add*: add more of a resource to the node's supply.

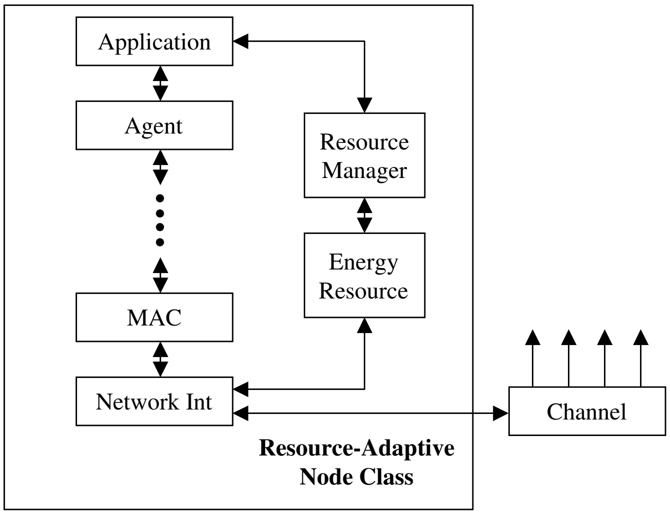


Figure A-2: Block diagram of a Resource-Adaptive Node.

- *remove*: remove some of a resource from the node's supply.
- *query*: find out what amount of the resource the node currently has.

For example, in this work, the Energy Resource is used to keep track of a node's energy. The initial energy level is set at the beginning of the simulation, and throughout the simulation, energy is removed by the Application as the node performs computation of data. The Network Interface can access the Energy Resource directly to remove energy during packet transmission and reception. The Application (and any lower-layer functions) can also find out how much energy remains at any given time. This is useful for implementing *resource-adaptive* protocols, that change their behavior based on the current level of the resource. The amount of energy removed for computation or communication is defined in Chapter 4. All of the protocols described in this dissertation were run on Resource-Adaptive Nodes.

A.2 Network Interface

The Network Interface performs the physical-layer functions. When it receives a packet from the MAC-layer, it sets the transmit power based on an approximation of the distance to the receiver (assuming power control is used, as in all the protocols described in this dissertation)[2], removes the

[2]If spread-spectrum is used, the number of bits in the packet is much higher (by the spreading rate) than the information bits. The transmit amplifier energy is per information bit. Therefore, the energy per transmitted bit is the energy per information bit divided by the amount of spreading (which is 1 if no spreading is used).

appropriate amount of energy to send the packet, and sends the packet onto the Channel. If the node has used up all its energy after transmitting the packet, the node is dead (nodeAlive = 0) and will be removed from the channel. Nodes that have died do not have any impact on the routing protocols, and any data sent to a node that is dead is thrown away. The following pseudo-code performs these functions:

```
NetworkInterface::transmit(packet)
{
  d = distance_to_receiver

  if  d < crossover_dist then
    Eamp = E_friss-amp * d^2
  else
    Eamp = E_two-ray-amp * d^4

  nbits = size_of_packet
  Epacket = nbits * Eelec + nbits * Eamp / spread_factor
  energy->remove(Epacket)
  if energy->query() < 0 then nodeAlive = 0

  channel->recv(packet)
}
```

When receiving data, the packet enters the node's Network Interface from the Channel. If the node is in the sleep state (nodeSleep = 1), the Network Interface discards the packet, since sleeping nodes can not receive or transmit any packets. Therefore, there is no energy cost to these nodes even when packets are being transmitted in their vicinity. However, if data are being sent to a sleeping node, they will be lost since the node has no way of knowing that it missed a packet. Therefore, the routing protocols must ensure that a transmitting node only sends data to a receiving node when the receiving node is awake to guarantee delivery of the data.

If the node is awake (nodeSleep = 0), the Network Interface determines the received power of the packet. If the received power is below a detection threshold ($P_{r-detect}$), the packet is thrown away, as the node would not have been able to detect that a packet was transmitted. If the received power is above the detection threshold but below a successful reception threshold ($P_{r-thresh}$), the packet is marked as erroneous and passed up the stack. It is not thrown away because reception of this packet affects the ability to successfully receive other packets at the same time. Finally, if the received power is above $P_{r-thresh}$, the packet has been received successfully and is passed up the stack to the MAC class. The following pseudo-code performs these functions:

```
NetworkInterface::recv(packet)
{
  if nodeSleep then
    discardPacket(packet)
    return

  Pr = ReceivePower(packet)

  if Pr < RXDetect then
    discardPacket(packet)
    return

  if Pr < RXThresh then packet_error = 1

  nbits = size_of_packet
  Epacket = nbits * Eelec
  energy->remove(Epacket)
  if energy->query() < 0 then nodeAlive = 0

  mac->recv(packet)

}
```

A.3 MAC Protocol

A new MAC protocol type, called MacSensor, has been created. This protocol is a combination of carrier-sense multiple access (CSMA), time-division multiple access (TDMA), and direct-sequence spread spectrum (DS-SS). The application determines which MAC protocol is used to send each message based on the constraints of the routing protocol. For example, in LEACH, if the packet is an advertisement or a join-request message, it is transmitted using a CSMA approach. If it is a data message being sent to the cluster-head, it is sent using a TDMA slot with the DS-SS code specified by the cluster-head. Using such an approach, the MAC protocol is always chosen such that it reduces energy dissipation by allowing nodes to remain in the sleep state for as long as possible (e.g., using TDMA) and minimizing collisions (e.g., using DS-SS on top of TDMA or using CSMA to reduce the number of collisions).

TDMA is implemented within the Application by only having the Application send data to the Agent during the specified TDMA time-slot[3]. CSMA is implemented in the MacSensor class,

[3]In this implementation of TDMA, it is assumed that the clocks of all the nodes are synchronized. However, it would be more accurate if the clock drift was modeled and explicit synchronization was performed. This is an area for future work.

and DS-SS is implemented jointly within the application and the MacSensor class. Non-persistent CSMA is used for the experiments. To perform CSMA, the node senses the channel before transmission. If the channel is currently being used by someone else, the node sets a back-off timer to expire after a random amount of time, where the timer is chosen uniformly with a maximum time equal to the transmit time of the packet it is waiting to transmit. This back-off policy for CSMA is effective because all nodes are transmitting packets with the same length during a given time. Therefore, the maximum amount of time that the channel will be busy is equal to the amount of time it would take to transmit the node's packet. Once the back-off timer expires, the node again senses the channel. If it is still busy (presumably someone else captured the channel first), the node again sets a back-off timer. This continues until the node senses a free channel. Once the channel is free, the TX_STATE variable is set to MAC_SEND and the node passes the packet onto the NetworkInterface. The node must also set a transmit-timer so it knows when it has finished transmitting the packet (and can reset the TX_STATE variable to MAC_IDLE). This is important because a node cannot transmit two packets at the same time, and a node cannot receive a packet while it is transmitting.

The Application is responsible for determining the pseudo-random noise sequence to use for DS-SS and performing data-spreading (modeled as an increase in data size). Since data are not actually transmitted in `ns`, the chosen spreading sequence is just listed in the header of the packet. The following pseudo-code performs these functions:

```
MacSensor::send(packet)
{
  if IamTransmitting || IamReceiving || channelBusy then
    time = random_number(0,TX_Time(packet))
    backOffTimer->start(packet, time)
    return

  TX_STATE = MAC_SEND

  TXTimer->start(TX_Time(packet))
  networkInterface->recv(packet)
}

MacSensorbackOffTimer::finish(packet)
{
  send(packet)
}
```

```
MacSensorTXTimer::finish()
{
  TX_STATE = MAC_IDLE
}
```

When MacSensor receives a packet from the Network Interface, it first determines whether the packet was sent using the spreading code which the node is currently receiving (listed in the packet header). This models the correlator stage of a spread-spectrum system, where packets sent using the node's pseudo-random noise sequence will correlate, whereas packets sent using some other pseudo-random noise sequence will not correlate and hence be dropped. However, the packet sill adds to the noise floor. If too many packets with different codes are being transmitted, all the packets will be received in error, since the noise floor will be too high to receive the packets sent with the correct code. Therefore, the node keeps track of which codes are being transmitted in its vicinity at all times. There is a maximum number of simultaneous transmissions that can occur using DS-SS (max_tx_at_once), based on the amount of spreading that occurs. As long as there are fewer than this number of transmissions heard at the receiving node, it is assumed that the reception of the packet is successful.

A node cannot receive a packet while transmitting. If the node is transmitting as a packet arrives, the received packet is marked as erroneous. However, it is not dropped because the receiver may be busy receiving this erroneous packet after the transmitter has finished. All packets that contain errors are dropped in the link-layer of the stack.

If the node is currently receiving another packet (pktRx) when this new packet (p) arrives, two situations can occur. First, pktRx might be sent with enough power to swamp out the reception of the new packet, p. In this case, capture occurs and p is dropped. On the other hand, if pktRx does not have enough power to swamp out p, both packets collide. In this case, the packet that will last the longest for reception is kept and marked as erroneous and the other packet is dropped. By keeping the packet that will last the longest, the receiver is busy for the maximum amount of time. Again, the packet that is kept will be dropped in the link-layer of the stack.

If the node is neither transmitting nor receiving when the new packet arrives, the RX_STATE variable is set to MAC_RECV and a timer is set that expires after the length of time required to receive the packet. When the timer expires, the node checks the address field of the packet. If the address is the node's address (or the broadcast address), the packet is sent up the stack to the queue. Otherwise, the packet is not intended for this node and is dropped.

```
MacSensor::recv(packet)
{
  pkt_code = get_code(packet)
  if pkt_code != myCode then
       discardPacket(packet)
       return

  if TX_STATE == MAC_SEND then packet_error = 1

  if RX_STATE == MAC_IDLE && TotalSigsIamHearing <= max_tx_at_once then
    RX_STATE = MAC_RECV
    time = TX_Time(packet)
    pktRx = packet
    RXTimer->start(packet, time)
  else
    if packet_error == 0 then packet_error = 1
    if ReceiverPower(packet) / ReceiverPower(pktRx) >= CPThresh then
      capture(packet)
    else
      collision(packet)
}

MacSensor::capture(packet)
{
  discardPacket(packet)
}

MacSensor::collision(packet)
{
  RX_STATE = MAC_COLL

  if TX_Time(packet) > RXTimer->expire then
    discardPacket(pktRx)
    pktRx = packet
    packet_error = 1
    RXTimer->start(packet,TX_Time(packet))
  else
    discardPacket(packet)
}

MacSensorRXTimer::finish(packet)
{
  RX_STATE = MAC_IDLE

  dst = GetDestinationAddr(packet)
  if  dst != MyAddr && dst != MAC_BROADCAST then discardPacket(packet)
  else queue->recv(packet)
}
```

A.4 LEACH Protocols

LEACH is implemented exactly as described in Chapter 3. Since LEACH is an application-specific protocol architecture, it is implemented as a subclass of `ns`'s Application class. LEACH-C is also implemented as described in Chapter 3. LEACH-C uses many of the same functions as LEACH (only the set-up phase differs), so it is implemented as a sub-class of LEACH.

A.5 Base Station Application

In our simulations, one node that is located away from the rest of the nodes is designated as the base station. This node has no energy constraints and is the node to which all data are eventually sent. Therefore, the base station node must keep track of all the data that it receives, as determining when the base station receives the data allows us to estimate the latency of different protocols and determining how much data is received during a given time allows us to determine the quality of different protocols. To perform these functions, an Application called BSApp was created.

For most of the protocols, this is the only function the base station serves. However, for LEACH-C, the base station must also receive small information packets from each node at the beginning of each round that contain the node's location and current energy level. Once the base station receives this data from all nodes, it must determine optimal clusters. As discussed in Section 3.5, the base station performs simulated annealing to determine these clusters. Since this is a computationally intense algorithm, the simulated annealing algorithm was implemented in C++, using an Agent called BSAgent. BSAgent determines the optimal clusters and sends this information up the stack to BSApplication. The base station node then broadcasts this information to the nodes in the network.

A.6 MTE Routing

For MTE routing, routes from each node to the base station were chosen such that each node's next-hop neighbor is the closest node that is in the direction of the base station. Each node requires 100 nJ to determine their next-hop neighbor[4]. When a node dies, all of that node's upstream neighbors (i.e., all the nodes that send their data to this node) begin transmitting their data to the node's next-hop neighbor. In this way, new routes do not need to be computed whenever a node dies.

[4]The amount of energy required to determine the initial routes is a parameter that can be easily varied.

Nodes adjust their transmit power to the minimum required to reach their next-hop neighbor. This reduces interference with other transmissions and reduces the nodes' energy dissipation. Communication with the next-hop neighbor occurs using a CSMA MAC protocol, and when collisions occur, the data are dropped. When a node receives data from one of its upstream neighbors, it forwards the data to its next-hop neighbor. This continues until the data reaches the base station.

A.7 Static-Clustering

The static clustering protocol implemented for the experiments is identical to LEACH except the clusters are chosen a-priori and fixed. The clusters are formed using the simulated annealing algorithm as in LEACH-C. Static clustering includes scheduled data transmissions from the cluster members to the cluster-head and data aggregation at the cluster-head.

A.8 Statistics Collection

We added statistics collection to keep track of the internal state of the network and the individual sensors during the simulation. At periodic intervals, the following data are collected:

1. Amount of energy consumed by each node.
2. Amount of data received at the base station from each node.
3. Number of nodes still alive.

Using these statistics, we can evaluate the effectiveness of the different communication protocols.

A.9 Simulation Parameters

The simulator uses the parameters shown in Table A.1.

Table A.1: Simulation parameters for the experiments described in this dissertation.

Parameter	Description
x	the maximum x location (m)
y	the maximum y location (m)
nn	the number of nodes in the simulation (including the base station)
rp	the routing protocol (either leach, leach-c, mte, or stat-clus)
stop	the length of the simulation (s)
eq_energy	1 if all nodes begin with equal energy, 0 otherwise
init_energy	the initial amount of energy given to each node (J)
num_clusters	the number of clusters desired for the protocols
bw	the radio bitrate (bps)
delay	the link-layer delay (s)
prop_speed	the channel propagation speed (m/s)
ll	the link-layer protocol
mac	the mac-layer protocol
ifq	the internal packet queue type
netif	the wireless channel
ant	the type of antenna
spreading	the amount of data-spreading for DS-SS
freq	the carrier frequency (Hz)
L	the system (non-propagation) loss
Gt	the Tx antenna gain
Gr	the Rx antenna gain
ht	the antenna height (m)
CSThresh	the threshold for detecting a packet (W)
RXThresh	the threshold for receiving an error-free packet (W)
EXcvr	the radio electronics energy (J/bit)
e_bf	the beamforming energy (J/bit/signal)
Efriss_amp	the transmit amplifier energy (J/bit/m^2)
Etwo_ray_amp	the transmit amplifier energy (J/bit/m^4)

Bibliography

[1] 3GPP Web Site. http://www.3gpp.org, 2000.

[2] M. Abbott and L. Peterson. Increasing Network Throughput by Integrating Protocol Layers. *IEEE/ACM Transactions on Networking*, 1(5):600–610, October 1993.

[3] W. Adjie-Winoto, E. Schwartz, H. Balakrishnan, and J. Lilley. The Design and Implementation of an Intentional Naming System. In *Proceedings of the 17th ACM SOSP*, December 1999.

[4] P. Agarwal and C. Procopiuc. Exact and Approximation Algorithms for Clustering. In *Proceedings of the Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 658–667, January 1999.

[5] J. Agre and L. Clare. An Integrated Architecture for Cooperative Sensing Networks. *Computer*, 33(5):106–108, May 2000.

[6] AWAIRS: Adaptive Wireless Arrays for Interactive Reconnaissance, Surveillance, and Target Acquisition in Small Unit Operations. http://www.janet.ucla.edu/awairs, 2000.

[7] D. Baker, A. Ephremides, and J. Flynn. The Design and Simulation of a Mobile Radio Network with Distributed Control. *IEEE Journal on Selected Areas in Communications*, 2(1):226–237, January 1984.

[8] H. Balakrishnan, V. Padmanabhan, S. Sesha, and R. Katz. A Comparison of Mechanisms for Improving TCP Performance over Wireless Links. In *IEEE/ACM Transactions on Networking*, December 1997.

[9] G. Begin and D. Haccoun. High-Rate Punctured Convolutional Codes for Viterbi and Sequential Decoding. *IEEE Transactions on Communications*, 37(11):1113–1125, November 1989.

[10] M. Bhardwaj and A. Chandrakasan. Power Aware Systems. *To be presented at the 34th Asilomar Conference on Signals, Systems and Computers*, November 2000.

[11] V. Bharghavan, A. Demers, S. Shenker, and L. Zhang. MACAW: A Media Access Protocol for Wireless LANs. In *Proceedings of SIGCOMM '94*, pages 212–225, September 1994.

[12] Bluetooth Project. http://www.bluetooth.com, 1999.

[13] G. Borriello and R. Want. Embedded Computation Meets the World Wide Web. *Communications of the ACM*, 43(5):59–66, May 2000.

[14] J. Broch, D. Maltz, D. Johnson, Y. Hu, and J. Jetcheva. A Performance Comparison of Multi-Hop Wireless Ad Hoc Netowrk Routing Protocols. In *Proc. 4th ACM International Conference on Mobile Computing and Networking (Mobicom'98)*, October 1998.

[15] R. Brooks and S. Iyengar. *Multi-Sensor Fusion*. Prentice-Hall, Inc., Upper Saddle River, NJ, 1998.

[16] M. Budagavi, W. Heinzelman, J. Webb, and R. Talluri. Wireless MPEG-4 Video Communication on DSP Chips. *IEEE Signal Processing Magazine*, 17(1):36–53, January 2000.

[17] Bult, K., et al. Low Power Systems for Wireless Microsensors. In *1996 International Symposium on Low Power Electronics and Design*, pages 17–21, August 1996.

[18] P. Chadwick. Sensitivity of Range in WLAN Receivers. In *IEE Colloquium on Radio LANs and MANs*, 1995.

[19] A. Chandrakasan, R. Amirtharajah, S.-H. Cho, J. Goodman, G. Konduri, J. Kulik, W. Rabiner, and A. Wang. Design Considerations for Distributed Microsensor Systems. In *IEEE 1999 Custom Integrated Circuits Conference (CICC)*, pages 279–286, May 1999.

[20] J. Chen, K. Sivalingam, P. Agrawal, and S. Kishore. A Comparison of MAC Protocols for Wireless Local Networks Based on Battery Power Consumption. In *Proceedings of IEEE INFOCOM '98*, April 1998.

[21] Y.-C. Cheng and T. Robertazzi. Communication and Computation Tradeoffs for a Network of Intelligent Sensors. In *Proceedings of the Computer Networking Symposium*, pages 152–161, April 1988.

[22] L. Clare, G. Pottie, and J. Agre. Self-Organizing Distributed Sensor Networks. In *SPIE Conference on Unattended Ground Sensor Technologies and Applications*, pages 229–237, April 1999.

[23] D. Clark and D. Tennenhouse. Architectural Considerations for a New Generation of Protocols. In *Proceedings of the ACM Symposium on Communications Architectures and Protocols*, pages 200–208, 1990.

[24] IEEE Computer Society LAN MAN Standards Committee. Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. *IEEE Std. 802.11-1997*, 1997.

[25] M. Dong, K. Yung, and W. Kaiser. Low Power Signal Processing Architectures for Network Microsensors. In *Proceedings 1997 International Symposium on Low Power Electronics and Design*, pages 173–177, August 1997.

[26] B. Erol, M. Gallant, G. Cote, and F. Kossentini. The H.263+ Video Coding Standard: Complexity and Performance. In *Proceedings of the 1998 Data Compression Conference (DCC '98)*, pages 259–268, April 1998.

[27] D. Estrin, R. Govindan, J. Heidemann, and S. Kumar. Next Century Challenges: Scalable Coordination in Sensor Networks. In *Proceedings of the Fifth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom '99)*, pages 263–270, August 1999.

[28] M. Ettus. System Capacity, Latency, and Power Consumption in Multihop-Routed SS-CDMA Wireless Networks. In *Radio and Wireless Conference (RAWCON '98)*, pages 55–58, August 1998.

[29] G. Foschini. Layered Space-Time Architecture for Wireless Communication in a Fading Environment When Using Multiple Antennas. In *Bell Laboratories Technical Journal*, volume 1, pages 41–59, 1996.

[30] J. Freiman. Protable Computer Power Sources. In *Proceedings of the Ninth Annual Battery Conference on Applications and Advances*, pages 152–158, January 1994.

[31] T. Furrer. Power Aware Embedded Operating System Design. Master's thesis, Massachusetts Institute of Technology, 2000.

[32] V. Garg and J. Wilkes. *Wireless and Personal Communications Systems.* Prentice Hall PTR, New Jersey, 1996.

[33] J. Goodman, A. Chandrakasan, and A. Dancy. Design and Implementation of a Scalable Encryption Processor with Embedded Variable DC/DC Converter. In *Proceedings of the 36th Design Automation Conference (DAC '99)*, pages 855–860, June 1999.

[34] ITU-T Recommendation H.223. Multiplexing Protocol for Low Bitrate Multimedia Communication. September 1997.

[35] ITU-T Recommendation H.223 and Annex A/Annex B/Annex C. Multiplexing Protocol for Low Bitrate Multimedia Communication over Low/Moderate/High Error Prone Channels. September 1997.

[36] J. Haartsen and S. Mattisson. BLUETOOTH—A New Low-Power Radio Interface Providing Short-Range Connectivity. *To appear: Proceedings of the IEEE Special Issue on Low-Power RF Systems*, 2000.

[37] J. Hagenauer. Rate-Compatible Punctured Convolutional Codes (RCPC Codes) and their Applications. *IEEE Transactions on Communications*, 36(4):389–400, April 1988.

[38] D. Hall. *Mathematical Techniques in Multisensor Data Fusion.* Artech House, Boston, MA, 1992.

[39] W. Heinzelman, M. Budagavi, and R. Talluri. Unequal Error Protection of MPEG-4 Compressed Video. In *Proceedings of the International Conference on Image Processing (ICIP '99)*, October 1999.

[40] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan. Energy-Efficient Routing Protocols for Wireless Microsensor Networks. In *Proc. 33rd Hawaii International Conference on System Sciences (HICSS '00)*, January 2000.

[41] W. Heinzelman, J. Kulik, and H. Balakrishnan. Adaptive Protocols for Information Dissemination in Wireless Sensor Networks. In *Proceedings of the Fifth Annual ACM/IEEE*

International Conference on Mobile Computing and Networking (MobiCom '99), pages 174–185, August 1999.

[42] W. Heinzelman, A. Sinha, A. Wang, and A. Chandrakasan. Energy-Scalable Algorithms and Protocols for Wireless Microsensor Networks. In *Proceedings of the International Conference Acoustics, Speech, and Signal Processing (ICASSP '00)*, June 2000.

[43] HomeRF Project. http://www.homrf.org, 1999.

[44] L. Hu. Distributed Code Assignments for CDMA Packet Radio Networks. *IEEE/ACM Transacations on Networking*, 1(6):668–677, December 1993.

[45] C. Huitema. *Routing in the Internet*. Prentice Hall, 1996.

[46] J. Inouye, S. Cen, C. Pu, and J. Walpole. System Support for Mobile Multimedia Applications. In *Proceedings of the IEEE 7th International Workshop on Network and Operating System Support for Digital Audio and Video*, pages 135–146, May 1997.

[47] Input/Output, Inc. http://www.i-o.com, 1999.

[48] C. Intanagonwiwat, R. Govindan, and D. Estrin. Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks. *To appear: Proc. 4th ACM International Conference on Mobile Computing and Networking (Mobicom'98)*, August 2000.

[49] P. Jain, C. Hutchinson, and T. Chanson. Protocol Architectures for Delivering Application-Specific Quality of Service. In *1995 International Conference on Network Protocols*, pages 313–320, November 1995.

[50] Japanese Developments in Lithion Ion Battery Technology. http://www.atip.or.jp/ATIP/public/atip.reports.94/li-ion.94.html, 2000.

[51] J. Kahn, R. Katz, and K. Pister. Mobile Networking for Smart Dust. In *Proceedings of the Fifth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom '99)*, August 1999.

[52] L. Klein. *Sensor and Data Fusion Concepts and Applications*. SPIE Optical Engr Press, WA, 1993.

[53] A. Kulkarni and G. Minden. Composing Protocol Frameworks for Active Wireless Networks. In *IEEE Communications Magazine*, pages 130–137, March 2000.

[54] T. Kwon and M. Gerla. Clutering with Power Control. In *Proceedings MILCOM '99*, volume 2, November 1999.

[55] J. Lansford and P. Bahl. The Design and Implementation of HomeRF: A Radio Frequency Wireless Networking Standard for the Connected Home. *To appear: Proceedings of the IEEE Special Issue on Low-Power RF Systems*, 2000.

[56] D. LeGall. MPEG: A Video Compression Standard for Multimedia Applications. *Communications of the ACM*, 34:46–58, April 1991.

[57] C. Lin and M. Gerla. Adaptive Clustering for Mobile Wireless Networks. *IEEE Journal on Selected Areas in Communications*, 15(7):1265–1275, September 1997.

[58] S. Lin and D. Costello. *Error Control Coding: Fundamentals and Applications.* Prentice-Hall, Inc., New Jersey, 1983.

[59] X. Lin and I. Stojmenovic. Power-Aware Routing in Ad Hoc Wireless Networks. In *SITE, University of Ottawa, TR-98-11*, December 1998.

[60] A. McDonald and T. Znati. A Mobility-Based Framework for Adaptive Clustering in Wireless Ad Hoc Networks. *IEEE Journal on Selected Areas in Communications*, 17(8):1466–1486, August 1999.

[61] T. Meng. Low-power GPS Reciever Design. In *Proceedings of the 1998 IEEE Workshop on Signal Processing Systems (SiPS '98)*, October 1998.

[62] T. Meng and R. Volkan. Distributed Network Protocols for Wireless Communication. In *Proc. IEEEE ISCAS*, May 1998.

[63] S. Meninger, J. Mur-Miranda, R. Amirtharajah, and A Chandrakasan. Vibration-to-Electric Energy Conversion. In *1999 International Symposium on Low-Power Electronic Design (ISLPED '99)*, pages 48–53, July 1999.

[64] R. Min, T. Furrer, and A Chandrakasan. Dynamic Voltage Scaling Techniques for Distributed Microsensor Networks. In *Workshop on VLSI (WVLSI '00)*, April 2000.

[65] MIT μ-AMPS Project. http://www-mtl.mit.edu/research/icsystems/uamps.html, 1999.

[66] T. Murata and H. Ishibuchi. Performance Evaluation of Genetic Algorithms for Flowshop Scheduling Problems. In *Proceedings of the 1st IEEE Conference on Evolutionary Computation*, volume 2, pages 812–817, June 1994.

[67] UCB/LBNL/VINT Network Simulator - ns. http://www-mash.cs.berkeley.edu/ns, 1998.

[68] ed. Oppenheim, A. *Applications of Digital Signal Processing.* Prentice-Hall, Inc., 1978.

[69] K. Pahlavan and A. Levesque. *Wireless Information Networks.* John Wiley & Sons, Inc., New York, 1995.

[70] S. Park and M. Srivastava. Power Aware Routing in Sensor Networks using Dynamic Source Routing. In *ACM MONET Special Issue on Energy Conserving Protocols in Wireless Networks*, 1999.

[71] C. Perkins and P. Bhagwat. Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers. In *Proceedings of the SIGCOMM '94 Conference on Communications Architectures, Protocols and Applications*, pages 234–244, August 1994.

[72] C. Perkins and E. Royer. Ad-Hoc On-Demand Distance Vector (AODV) Routing. In *Proceedings of the Second IEEE Workshop on Mobile Computing Systems and Applications (WMCSA '99)*, pages 90–100, 1999.

[73] G. Pottie. Wireless Sensor Networks. In *Information Theory Workshop, 1998*, pages 139–140, June 1998.

[74] G. Pottie and W. Kaiser. Wireless Integrated Network Sensors. *Communications of the ACM*, 43(5):51–58, May 2000.

[75] T. Rappaport. *Wireless Communications: Principles & Practice.* Prentice-Hall, Inc., New Jersey, 1996.

[76] J. Razavilar, F. Rashid-Farrokhi, and K. Liu. Software Radio Architecture with Smart Antennas: A Tutorial on Algorithms and Complexity. *IEEE Journal on Selected Areas in Communications*, 17(4):662–676, April 1999.

[77] L. Robinson. Japan's new mobile broadcast company: Multimedia for cars, trains, and handhelds. In *Advanced Imaging*, pages 18–22, July 1998.

[78] R. Ruppe, S. Griswald, P. Walsh, and R. Martin. Near Term Digital Radio (NTDR) System. In *Proceedings MILCOM '97*, volume 3, pages 1282–1287, November 1997.

[79] SCADDS: Scalable Coordination Architectures for Deeply Distributed Systems. http://netweb.usc.edu/SCADDS, 2000.

[80] G. Schuster. Stanford Mathematical Geophysics Summer School Lectures: Basics of Exploration Seismology and Tomography. http://utam.geophys.utah.edu/stanford/ch.html, January 1999.

[81] K. Scott and N. Bambos. The Self-Organizing Wireless Adaptive Network (SWAN) Protocol for Communication Among Mobile Users. In *Proceedings of the IEEE Globecom '95*, 1995.

[82] K. Scott and N. Bambos. Routing and Channel Assignment for Low Power Transmission in PCS. In *5th IEEE Int. Conf. on Universal Personal Communications*, volume 2, pages 498–502, September 1996.

[83] S. Sheng, L. Lynn, J. Peroulas, K. Stone, I. O'Donnell, and R. Brodersen. A Low Power CMOS Chipset for Spread Spectrum Communications. In *Proceedings of the 42nd Solid-State Circuits Conference (ISSCC '96)*, February 1996.

[84] T. Shepard. A Channel Access Scheme for Large Dense Packet Radio Networks. In *Proc. ACM SIGCOMM*, pages 219–230, August 1996.

[85] S. Singh, M. Woo, and C.S. Raghavendra. Power-Aware Routing in Mobile Ad Hoc Networks. In *Proceedings of the Fourth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom '98)*, October 1998.

[86] A. Sinha, R. Min, T. Furrer, and A Chandrakasan. Operating System Directed Power Management in Wireless Sensor Networks. *Submitted to: Ninth International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS-IX)*, November 2000.

[87] Smart Dust: Autonomous Sensing and Communication in a Cubic Millimeter. http://robotics.eecs.berkeley.edu/~pister/SmartDust, 2000.

[88] D. Smithgall. Toward the 60 gm Wireless Phone. In *Proceedings of the 1998 Radio and Wireless Conference (RAWCON '98)*, pages 157–159, August 1998.

[89] Y. Takishima, M Wada, and H. Murakami. Reversible Variable Length Codes. *IEEE Transactions on Communications*, 43(2):158–162, February 1995.

[90] R. Talluri. Error-Resilient Video Coding in the ISO MPEG-4 Standard. In *IEEE Communications Magazine*, pages 2–10, June 1998.

[91] D. Tennenhouse and D. Wetherall. Towards an Active Network Architecture. In *Computer Communication Review*, 1996.

[92] J. Vainio, H. Mikkola, K. Jarvinen, and P. Haavisto. GSM EFR-Based Multi-Rate Codec Family. In *Proceedings of the 1998 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '98)*, volume 1, pages 141 – 144, May 1998.

[93] P. Varshney. *Distributed Detection and Data Fusion.* Springer-Verlag, New York, 1997.

[94] A. Wang, W. Heinzelman, and A. Chandrakasan. Energy-Scalable Protocols for Battery-Operated Microsensor Networks. In *Proceedings of the 1999 IEEE Workshop on Signal Processing Systems (SiPS '99)*, pages 483–492, October 1999.

[95] M. Weiser. Hot Topics: Ubiquitous Computing. In *IEEE Computer*, October 1993.

[96] L. Williams and L. Emergy. Near Term Digital Radio- a First Look. In *Proceedings of the 1996 Tactical Communications Conference*, pages 423–425, April 1996.

[97] WINS: Wireless Integrated Network Sensors. http://www.janet.ucla.edu/WINS, 2000.

[98] Wireless Today: Forecast of Users with Wireline Versus Wireless High-Speed (Individual) Access, North America, 1998-2003. http://www.wirelesstoday.com/snaparchives, January 1999.

[99] Wireless Today: World Cellular Subscribers. http://www.wirelesstoday.com/snaparchives, September 1999.

[100] Wireless Today: Carriers Take Many Paths to 3G Deployment. http://www.wirelesstoday.com/snaparchives, April 2000.

[101] K. Yao, R. Hudson, C. Reed, D. Chen, and F. Lorenzelli. Blind Beamforming on a Randomly Distributed Sensor Array System. In *Proceedings of the 1998 IEEE Workshop on Signal Processing Systems (SiPS '98)*, October 1998.

[102] C. Yung, H. Fu, C. Tsui, R. Cheng, and D. George. Unequal Error Protection for Wireless Transmission of MPEG Audio. In *Proceedings of the 1999 IEEE International Symposium on Circuits and Systems (ISCAS '99)*, volume 6, pages 342–345, June 1999.

[103] H. Zimmermann. OSI Reference Model–The ISO Model of Architecture for Open Systems Interconnection. *IEEE Transactions on Communications*, 28(4), April 1980.