# ENERGY LATENCY TRADEOFFS FOR MEDIUM ACCESS AND SLEEP SCHEDULING IN WIRELESS SENSOR NETWORKS

by

Gang Lu

---

A Dissertation Presented to the
FACULTY OF THE GRADUATE SCHOOL
UNIVERSITY OF SOUTHERN CALIFORNIA
In Partial Fulfillment of the
Requirements for the Degree
DOCTOR OF PHILOSOPHY
(Electrical Engineering)

December 2005

# Dedication

To my dear wife and parents,

For your love and support!

# Acknowledgements

First, I am deeply grateful for my advisor, Prof. Bhaskar Krishnamachari. He is my mentor not only in career, but also in life and personality. I thank him for his guidance and support.

I would like to thank other members on my defense and qualify committee including Prof. Raghavendra, Prof. Ghandeharizadeh, Prof. Helmy and Prof. Psounis, for their useful feedback during my qualify exam and defense. Especially, I would like to thank Prof. Cauligi Raghavendra, under which I spent my first 3 years in USC, for his generous guidance and support.

I have been working in both Prof. Krishnamachari's Autonomous Network Research Group (ANRG) and Prof. Raghavendra's group. Both groups are excellent groups, in terms of both academic and life. I feel really lucky to be a member of the groups and would like to thank all the members in the group, including Dongjin Son, Thrasyvouls Spyropulos(Akis), Avinash Sridharn, Rahul Urgaonkar, Kiran Yedavli, Yang Yu, Narayanan Sadagopan, Marco Zuniga, Sundeep Pattem, Shyam Kapadia, Caimu Tang, Eric Coe. I would like to thank Yang Yu and Narayanan Sadagopan specially for the insightful discussions I have with them.

# Table of Contents

# List of Tables

# List of Figures

# Abstract

Wireless sensor networks are expected to be used in a wide range of applications from environment monitoring to event detection. The key challenge is to provide energy efficient communication; however, latency remains an important concern for many applications that require fast response.

*The central thesis of this work is that energy efficient medium access and sleep scheduling mechanisms can be designed without necessarily sacrificing application-specific latency performance*. We validate this thesis through results from four case studies that cover various aspects of medium access and sleep scheduling design in wireless sensor networks.

Our first effort, DMAC, is to design an adaptive low latency and energy efficient MAC for data gathering to reduce the sleep latency. We propose staggered schedule, duty cycle adaptation, data prediction and the use of more-to-send packets to enable seamless packet forwarding under varying traffic load and channel contentions. Simulation and experimental results show significant energy savings and latency reduction while ensuring high data reliability.

The second research effort, DESS, investigates the problem of designing sleep schedules in arbitrary network communication topologies to minimize the worst case end-to-end latency (referred to as delay diameter). We develop a novel graph-theoretical formulation, derive and analyze optimal solutions for the tree and ring topologies and heuristics for arbitrary topologies.

The third study addresses the problem of minimum latency joint scheduling and routing (MLSR). By constructing a novel delay graph, the optimal joint scheduling and routing can be solved by $M$ node-disjoint paths algorithm under multiple channel model. We further extended the algorithm to handle dynamic traffic changes and topology changes. A heuristic solution is proposed for MLSR under single channel interference.

In the fourth study, EEJSPC, we first formulate a fundamental optimization problem that provides tunable energy-latency-throughput tradeoffs with joint scheduling and power control and present both exponential and polynomial complexity solutions. Then we investigate the problem of minimizing

total transmission energy while satisfying transmission requests within a latency bound, and present an iterative approach which converges rapidly to the optimal parameter settings.

# Chapter 1

# Introduction

## 1.1 Overview

A wireless sensor network is a distributed sensing network comprised of thousands, or even tens of thousands small devices that sense, collect and disseminate information about the environment [1, 104, 103]. With each node equipped with radios of wireless communication capability and sensors that can sense certain physical phenomena such as acoustics, light, temperature, humidity and vibrations, wireless sensor networks (WSN) enable a wide range of applications, such as target tracking [72], habitat sensing [106, 105] and fire detection. The capability of sensor nodes [35, 71, 72] are very different from traditional nodes in computer networks. These devices have very limited energy, processing power, storage, communication range and rate. For example, in a Mote [35], the processor is Atmega 128L [2], a low-power micro-controller; its memory is less than 1MB. The radio used in MICA2 Mote is CC1000 [3], with radio rate of only 38.4Kbaud and range less than 500 feet range.

Wireless sensor networks require a new set of protocol stacks because of new features of wireless sensor networks [1, 104, 103]. First, most nodes in sensor networks are likely to be battery powered and it is not feasible to recharge or replace the batteries. Second, sensor networks are in large scale with hundreds or even thousands nodes randomly deployed in an ad hoc fashion with little human management [83, 87]. Third, the traffic pattern in sensor networks varies with different sensor network applications. Major traffic could be in-network local communication or from sensors to a common sink in a tree topology [96, 92].

There are many research challenges for the development of a wireless sensor network. First, Energy efficiency is a critical design issue in wireless sensor networks in order to prolong the network lifetime. Measurements show that wireless radio consumes a significant amount of energy [4, 5]. Hence energy efficient communication protocols are required for the success of wireless sensor network technology.

| | Computation<br>6 FFT<br>1 Beamforming | Communication | $\frac{E_{comm}}{E_{comp}}$ |
|---|---|---|---|
| 59MHz<br>0.83V | 2154.5uJ | 68576.9uJ | 31.8 |
| 133MHz<br>1.1V | 3635.8uJ | | 18.9 |

Figure 1.1: Power specification of some radios.

| | Computation<br>6 FFT<br>1 Beamforming | Communication | $\frac{E_{comm}}{E_{comp}}$ |
|---|---|---|---|
| 59MHz<br>0.83V | 2154.5uJ | 68576.9uJ | 31.8 |
| 133MHz<br>1.1V | 3635.8uJ | | 18.9 |

Figure 1.2: Energy cost for communication and computation. The energy cost for computation is calculated by JouleTrack [90]. Communication cost is based on ORINOCO WLAN card [82].

Latency remains an important concern for many event-driven applications such as fire detection, environment surveillance. Depending on application scenarios, packets may be required to be delivered either as soon as possible, or within a predefined latency bound.

As the basic communication building block, medium access and sleep scheduling protocols are crucial in deciding the energy and latency performance of the network. Previous works on MAC protocols in this domain generally either provide energy efficiency at the cost of high latency or provide low latency at the cost of energy. These observations motivate us to analyze the inherent tradeoff and design energy efficient and low latency medium access and sleep scheduling algorithms suitable for sensor networks.

## 1.2 Sensor Network MAC Protocol

### 1.2.1 Performance Metrics

Typically in WSNs, nodes coordinate locally to perform data processing and deliver messages to a common sink. The desired design features for medium access control protocols in a WSN are:

1. **Self-organization:** In many envisioned scenarios, the sensor deployment distribution will be very dense, in order to provide higher accuracy and fine-grained information about the environment and also because a larger aggregate amount of energy is available in a dense deployment. Because of the environment and large scale of nodes, the nodes are usually randomly deployed and there can be little human management. Thus the MAC protocol must be able to self-organize the communication infrastructure for data transfer.

2. **Energy efficiency:** Sensor nodes operate on battery and it is often not feasible to replace or recharge batteries for sensor nodes. Energy efficiency is a critical issue in order to prolong network lifetime. Measurements show that wireless radio consumes a significant amount of energy [4, 5]. Figure 1.1 shows the power specification of some current radios. Figure 1.2 shows that the communication energy cost can be much higher than computation energy cost. In particular, MAC protocols must minimize the radio energy costs in sensor nodes.

3. **Low latency:** Latency requirements depend on the application. In target tracking applications [72], an event detected needs to be reported to a sink in real time so that appropriate action can be taken promptly. In other applications, sensor nodes can store data in the network waiting for sink to query and latency is not an issue.

4. **High throughput:** Throughput requirements vary with different applications too. Some applications need to sample the environment with fine temporal resolution. In such applications, the more data the sink receives the better. In other applications, such as fire detection, it may suffice for a single report to arrive at the sink. Typically in sensor network applications such as fire detection, traffic may be light most of the time; when the environment changes abruptly due to a significant event (fire detected), for a short period the traffic may be very intense [1, 11]. MAC protocols should be able to handle both cases efficiently with high delivery ratio.

5. **Fairness:** Fairness requirements depends on the application too. In many applications, particularly when bandwidth is scarce, it is important to ensure that the sink receives information from

all sources in a fair manner [79]. In other applications, per-node MAC level fairness is not important as long as application-level performance is not degraded [60]. For example, the sink can only process data after receiving all packets from two sensors, which can let one node send all of its packets first, then let the second node transmit. The performance in application level is same.

6. **Reliability and Robustness:** Because of the harsh channel quality, frequent nodes failures and dynamic topology changes, the MAC protocols must be robust and reliable to enable efficient communication.

7. **Scalability** MAC protocols must be able to handle large networks with dynamic changes since wireless sensor networks can have hundreds or even thousands of nodes [1].

Among these important requirements for MACs, energy efficiency is typically the primary goal in WSN. It is often difficult to achieve good performance on all the above features for a MAC protocol. It is important to trade off secondary requirements to the most important factors when designing a MAC protocol for a specific sensor network application.

### 1.2.2 Taxonomy

Generally there are two kinds of MAC protocols in wireless sensor networks: contention-based and schedule-based MAC. The typical contention-based MAC is the standardized IEEE 802.11 distributed coordination function [78]. It is very successful in current wireless LAN market because of its simplicity and robustness. However it is not energy efficient because the nodes spend a lot of time in idle listening mode, which consumes similar energy as the receiving mode [4, 5]. There are two important components in contention-based MAC: the listening mechanism and the backoff scheme [79]. The listening time can be random or constant. There are four choices for backoff mechanism: no backoff, fixed window, exponential increase and exponential decrease. Simulation results in [79] show that the constant listening periods are energy-efficient. However, the traffic in sensor network can be highly synchronized, so a random delay is introduced before transmission to make it robust against repeated collisions. A contention-based MAC protocol has to monitor the channel activity before transmission. If the channel is busy, the sender has to back off a random period. In the backoff period, the node also monitors the channel to decide whether it can decrease the backoff timer. So the node is in idle listening mode which consume energy. Contention and collision could result wasted packet transmissions which are waste of energy. There are also overheads due to control packets to reduce contention. To save energy, the key idea is to turn off the radio when a node does not participate in data communication.

Table 1.1: Comparison of schedule and contention-based MAC protocols

| MAC | Self-organization | Energy | Latency | Throughput |
|---|---|---|---|---|
| Schedule-based | poor | good | poor | medium |
| Contention-based | good | poor | medium | medium |
| MAC | Fairness | Robust | Scalability | |
| Schedule-based | good | poor | poor | |
| Contention-based | medium | good | good | |

The advantage of contention-based MAC is its simplicity and robust which is very useful in wireless sensor networks.

Schedule-based MAC protocols rely on channel reservation. A typical schedule-based MAC is TDMA [10, 67, 93]. It is straightforward to employ energy efficient techniques in schedule-based MAC since a node can turn on its radio only in its reserved time slot and turn off radio in other slots. There is also no overhead due to contention and collision. However, schedule-based MAC protocols need synchronization which may pose significant overhead. Schedule-based MAC is more complicated than contention-based MAC, and is worse in terms of self-organization. When there is change in network topology, the schedule has to been adjusted which results in poor scalability and less robustness. Another disadvantage of schedule-based MAC is its possible low channel utilization. A node is fixed to communicate in its reserved slot which means a node has only limited channel capacity while another node may waste its reserved slot when it has nothing to send/receive. This will also result in low throughput and high latency.

Because contention-based and schedule-based MAC both have advantages and disadvantages, there are efforts to combine them together, such as MAC in IEEE 802.15.4. In IEEE 802.15.4, a superframe is divided into a Contention Access Period (CAP) and a Contention Free Period (CFP). A node can decide to use either CAP or CFP based on the requirement of its data communication. Detail will be discussed in next section.

Table 1.1 summarizes the comparison of schedule and contention-based MAC using the performance metrics.

## 1.3 Energy Latency Tradeoffs in Medium Access Control

### 1.3.1 Energy Consumption Sources

We first identify the following major sources of energy waste.

1. Collision: when a transmitted packet is corrupted it has to be discarded, and the follow-on re-transmissions increase energy consumption. Collision increases latency as well.

2. Overhearing: when a node receives packets that are destined to other nodes.

3. Control packet overhead: sending and receiving control packets consumes energy too, and less useful data packets can be transmitted.

4. Idle listening: listening to receive possible traffic that is not sent. This is especially of concern in many sensor network applications. If nothing is sensed, nodes are in idle mode for most of the time. However, in many MAC protocols such as IEEE 802.11 or CDMA nodes must listen to the channel to receive possible traffic. Many measurements have shown that idle listening consumes 50% to 100% of the energy required for receiving [4, 5]. For example, Stemm and Katz measured that the idle:receive:send ratios are 1:1.05:1.4, while the Digitan 2 Mbps Wireless LAN module (IEEE 802.11/2Mbps) specification shows idle:receive:send ratios is 1:2:2.5 [5].

5. Transmission power: Some radios have fixed transmission power or MAC protocols do not utilize the multiple transmission powers. When two nodes are nearby, the necessary transmission power may be much smaller than the maximal transmission power of the radio. If the radio only uses the maximal power to transmit, it is a significant energy wastage.

### 1.3.2 MAC Energy Saving Techniques

Corresponding to the source of energy wastage, there are several energy saving techniques in MAC layer:

1. **Low duty cycle:** In order to reduce the energy wastage due to idle listening, a node turns on its radio periodically to see if there is communication request, and goes back to sleep afterwards.

2. **TDMA link scheduling:** In TDMA link scheduling, a node can turn on its radio only in its reserved time slot and turn off radio in other slots. There is also no overhead due to contention and collision.

3. **Power control:** A node could have different distance to its neighbor nodes. It is not necessary to always use the maximum transmission power to reach the nearby nodes, as long as the $SINR$ requirement is satisfied at the receiver node. Many existing radios can support multiple transmission power. For example, CC1000 used in MICA2 motes can support 31 different transmission powers. Power control techniques have been widely used in topology control in wireless networks. Recently there are works on jointly scheduling and power control to save energy.

### 1.3.3 Energy-Latency Tradeoffs

Previous works (in particular [60], [26], [62],[28], [63], [33], [66]) have identified idle listening as a major source of energy wastage. To design an energy efficient MAC, it is essential to turn the radio off when a node does not participate in any data delivery. However, a node that is sleeping is no longer part of the network, and thus cannot help to deliver the sensor data from its neighbors to its destination. When a node has a packet for its neighbor which is in asleep, it has to wait until its neighbor is active. This creates a fundamental trade-off between energy and latency.

## 1.4 Research Contributions

Contention-based MAC protocols are particularly suitable for applications where the traffic load is unpredictable or varying rapidly, because of their low overhead requirements and flexibility. The SMAC protocol [60] proposed a low duty cycle scheme which puts the radio to sleep periodically (sleep schedule) to save energy. However, this approach increases the packet delivery latency significantly due to synchronized duty cycles for all nodes. In the first two studies, DMAC and DESS, we investigate how to minimize latency in such sleep scheduled contention-based MAC protocols without increasing the energy cost.

### 1.4.1 Data-Gathering MAC (DMAC) in a tree

DMAC is an adaptive low latency and energy efficient MAC for data gathering in wireless sensor networks [46, 47]. In this study, we first show the *Data Forwarding Interruption Problem* in synchronized sleep schedule protocols, whereby not all nodes on a multihop path are notified of data delivery in progress, resulting in significant sleep delay. Then we propose a staggered schedule for tree-based data gathering to solve the interruption problem by giving the sleep schedule of a node an offset that depends upon its depth on the tree. DMAC also adjusts the duty cycles adaptively to keep the latency low

Table 1.2: Application scenarios for the DMAC, DESS, MLSR and EEJSPC studies

| Study | DMAC | DESS | MLSR | EEJSPC |
|---|---|---|---|---|
| Access Method | Contention-based | Contention-based | Contention-free | Contention-free |
| Traffic | Unpredictable | Light traffic | Predictable, Long lived | Predictable, low dynamics |
| Topology | Data gathering tree | Any to any communication | multiple sinks Data gathering | Arbitrary topology |
| Energy Efficiency | By sleep scheduling | By sleep scheduling | By sleep scheduling | Besides sleep, by power control |
| Source of Latency | sleep schedule | sleep schedule | sleep schedule | link schedule |
| Latency objective | Minimizing average latency (best effort) | Minimizing latency diameter | Minimizing average latency | Maintaining per hop latency bound |

even under varying traffic load conditions. We further propose a data prediction mechanism and the use of more-to-send packets in order to alleviate problems pertaining to channel contention and collisions. Simulation and experimental results show that DMAC provides significant energy savings and latency reduction while ensuring high data reliability.

### 1.4.2 Delay Efficient Sleep Scheduling (DESS) in arbitrary network

In the second study, called DESS, we take an algorithmic approach and study a more general version of the problem: *how should the sleep schedule be designed in arbitrary network communication topologies, in order to minimize the worst case end-to-end latency while providing energy efficiency through periodic sleep?* We develop a novel graph-theoretical formulation of the problem and prove that minimizing the worst-case end-to-end communication latency (referred to as the delay diameter) is in general NP-hard. However, we are able to derive and analyze optimal solutions for the tree and ring topologies. Several heuristics for arbitrary topologies are also proposed and evaluated by simulations. Our simulations suggest that distributed heuristics may perform poorly because of the global nature of the constraints involved.

### 1.4.3 Minimum Latency Joint Scheduling and Routing (MLSR)

In the third study we address the important problem of minimizing communication latency while providing energy-efficiency for nodes in wireless sensor networks. Different from DESS, where the objective is to minimize the worst case latency given the fixed duty cycling requirement for each sensor, in MLSR the interest is in the average latency for only the current active flows. As the flows in some wireless

sensor network can be long-lived and predictable, it is possible to design schedules for sensor nodes so that nodes can wake up only when it is necessary and asleep during other times. The routing layer decision is also closed coupled to the wakeup/sleep schedule of the sensor nodes. We formulated a joint scheduling and routing problem with the objective of finding the schedule and route for current active flows with minimum average latency. By constructing a novel delay graph, the problem can be solved optimally by using a $M$ node-disjoint paths algorithm under a FDMA channel model. We further extend the algorithm to handle dynamic traffic changes and topology changes in wireless sensor networks. We also propose a heuristic solution for the minimum latency joint scheduling and routing problem under single channel interference.

### 1.4.4 Energy Efficient Joint Link Scheduling and Power Control

TDMA-based contention-free medium access protocols [6, 7, 8, 12, 9, 10, 13] can be more energy efficient than random access, particularly when traffic is predictable or slowly changing. In most prior studies of TDMA-scheduling, typically a simple model for interference is used where a receiving node sees interference from another transmitter if and only if it is within some nominal range $R_I$. This model, while useful in providing a simple graph-coloring approach to TDMA scheduling, can be quite misleading in practice for two reasons. First, simultaneous wireless transmissions within the nominal range do not necessarily collide if the signal to interference plus noise ratios (SINR) at the corresponding receivers are sufficiently high. Second, aggregate interference from multiple out-of-range transmitters can be high enough to cause collisions. Another concern with many studies of TDMA in wireless ad hoc and sensor networks is that they ignore the possibility of variable transmission power. In practical systems this can be an important tunable parameter for reliable and energy-efficient communication, because higher transmit powers can increase the SINR at the receiver to enable successful reception on a link, and lower transmission power can mitigate interference to other simultaneously utilized links. In the third study, TJSPC [50], we study the problem of TDMA link scheduling using a realistic SINR-based interference model, explicitly taking transmission power control into account.

In the fourth study, we investigate the problem of energy efficiency in TDMA link scheduling with transmission power control using a realistic SINR-based interference model, given packets of a set of links to be transmitted within a latency bound. First, we formulate a fundamental optimization problem (TJSPC) that provides tunable tradeoffs between energy, throughput and latency through a single parameter $\beta$. We present both exponential and polynomial complexity solutions to this problem and evaluate their performance. Our results show that for moderate traffic loads, with appropriate tuning

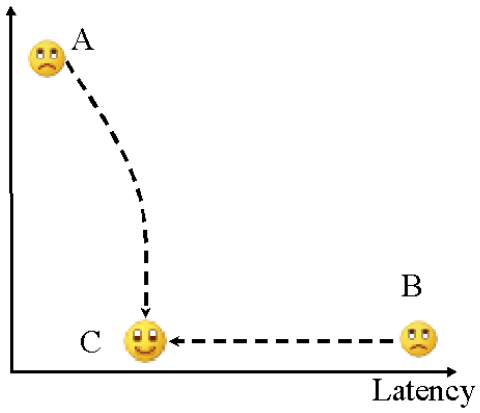



Figure 1.3: Energy-latency tradeoffs

of parameters, major energy savings can be obtained without significantly sacrificing throughput. We then investigate the scheduling and power control problem with the objective of minimizing the total transmission energy cost under the constraint that all transmission requests are satisfied (JSPC-TR). We present an iterative approach to solve JSPC-TR that leverages the heuristics for TJSPC and converges rapidly to the setting of $\beta$ which achieves energy efficiency while guaranteeing data delivery.

### 1.4.5 Summary

Table 1.2 shows the different application scenarios of DMAC, DESS, MLSR and EEJSPC, the four schemes we presented in this thesis, allowing for a high-level comparison between these studies. Both DMAC and DESS are studies pertaining to contention-based MAC protocols while MLSR and EEJSPC are contention-free schedule-based MAC. DMAC assumes a data gathering tree topology with unpredictable or fast changing traffic load. DESS, however, assumes a light traffic load with an arbitrary any-to-any communication model. MLSR assumes predictable long-lived traffic load in multiple-sinks data gathering scenario. In EEJSPC, although communication can on any arbitrary topology, the traffic load is expected to be more predictable. DMAC, DESS and MLSR provide energy efficiency only by putting nodes into sleep. In EEJSPC, besides the energy saving by sleeping during inactive slots, the focus is reduction of energy during packet transmissions in the active slots through the power control algorithm. The latency in DMAC, DESS and MLSR is due to the sleep schedule while in EEJSPC the latency is caused by the TDMA-link schedule. In DMAC, MLSR and DESS, the goal is to minimize the latency while keeping energy efficiency at a predetermined level. In EEJSPC, however, the goal is to minimize the energy cost subject to a latency bound constraint.

Figure 1.3 is an illustration placing the contributions of DMAC, DESS, MLSR and EEJSPC in context. It shows different operating points on an energy-versus-latency graph. Previous works generally either operate at point A, which represents a reduction of latency by sacrificing energy (e.g. prior works on joint link scheduling and power control), or at point B, which sacrifices latency to reduce energy (e.g. prior works on sleep scheduled MAC protocols). Our goal in this thesis is to show that MAC protocols can be carefully designed in many situations to operate at point C, which provides energy efficiency without necessarily suffering high latency – DMAC, DESS, MLSR and EEJSPC exemplify such an operating point.

# Chapter 2

# Related Works

A lot of MAC protocols and scheduling algorithms have been proposed in the last few years for wireless sensor networks with different sensor network application assumptions. This chapter provides a survey of the current state of the art of MAC protocols with an emphasis on energy efficiency. We also classify and compare different schemes with several performance metrics. Then we briefly describe the works that are closely related to this thesis.

## 2.1 Energy Efficient MAC

### 2.1.1 Energy Efficient Contention-based MAC

PAMAS [33] is one of the earliest contention-based energy efficient MAC protocols. The goal of PAMAS is to reduce the energy cost in overhearing among neighboring nodes. When a data transmission is in process between two nodes, all nearby nodes overhear the packet and can not send or receive. Energy saving can be achieved to put those nearby nodes to sleep. PAMAS protocol is based on MACA protocol and modified to provide separate channels for RTS/CTS control packets and data packets. Channel monitoring is on the control channel. The basic technique is that the receiving node transmit a busy tone over the control channel to indicate that the data channel is busy. The use of a separate control channel allows a node to determine when and for how long to power off the radio through the use of a *probe* protocol. Theoretical analysis is also presented in [33]. Simulation results show that 10% to 70% power savings can be achieved for fully connected topologies.

SMAC [60] tries to reduce all four energy waste with three techniques: periodic listening and sleep, collision and overhearing avoidance and message passing. By periodic listening and sleep, S-MAC [60] reduces the duty cycle of a sensor node by operate nodes in a periodic active/sleep schedule. During

sleep periods, nodes turn off radio completely to conserve energy. During active periods, nodes turn on radio to Tx/Rx messages. This will reduce channel capacity, however low traffic load is expected during most of the sensor network lifetime. So by default, nodes operate on low-duty-cycle mode to save energy. This, however, sacrifices latency. SMAC further argues that in sensor network, per-hop MAC level fairness is not an important issue since all nodes cooperates for a single common task. By trading off per-node fairness to efficiently transmit long messages, SMAC divides long message into several small fragments and transmit them in a burst. Only one pair of RTS/CTS is used to reserve the medium for transmitting all small fragments. Fragments are retransmitted separately in case of bit error. By message passing, control overhead is reduced and the high cost of retransmitting a long packet is avoided. Similar to PAMAS, SMAC avoids overhearing by putting all immediate neighbors of both the sender and the receiver to sleep after they hear an RTS or CTS packet. NAV is maintained at each node to indicate the activity in its neighborhood. A node should sleep until its NAV is zero.

Although a low duty cycle MAC is energy efficient, it has three side-effects. First, it increases the packet delivery latency. An intermediate node may have to wait until the receiver wakes up before it can forward a packet. This is called *sleep latency* in SMAC [60], which increases proportionally with hop length by a slope of schedule length (active period plus sleep period). Second, a fixed duty cycle does not adapt to the traffic variation in sensor network. A fixed duty cycle for the highest traffic load results in significant energy wastage when traffic is low while a duty cycle for low traffic load results in low message delivery and long queuing delay. Third, a fixed synchronous duty cycle may increase the possibility of collision. If neighboring nodes turn to active state at the same time, all may contend for the channel, making a collision very likely.

To solve the fixed duty cycle problem in SMAC, TMAC [26] proposes schemes to adjust duty cycle according to current traffic load. The basic idea of TMAC is that a node will keep active until no *activation event* has occurred for a time *TA*. An activation event includes: the firing of a periodic frame timer, the reception of any data on the radio, the sensing of communication on the radio, etc [26]. A node will sleep if its not in an active period. The novel idea of TMAC is to transmit all messages in a burst in the active period so that idle listening is reduced. The authors of TMAC identified the *early sleeping* problem, in which a node goes to sleep when a neighbor still has message for it. TMAC employs FRTS to solve the problem. A node overhearing a CTS packet destined for another node sends a FRTS packet immediately. A node that receives an FRTS packet will wake up after the duration information in FRTS and be ready to receive packets. Another solution proposed by TMAC is named “Taking priority on full buffers”. When a node’s transmit/routing buffers are almost full, it prefers sending to receiving by

immediately sends its own RTS packet to another node, instead of replying with a CTS. The probability of early sleeping problem occurs is reduced.

In scenarios where minimizing sleep latency is not important (non time critical applications), [36] presents an analysis on bounds on the delay of sending data from a node to a sink using a completely decentralized duty cycling scheme. The authors show that if each sensor turns on and off independent of the other sensors, the delay incurred is proportional to the distance of the node from the sink. However the rate of this linear increase is not dependent on the locations of the nodes, but on the node density, transmission range and the average active and sleep durations.

The question arises whether energy-efficient duty cycling may be maintained while reducing sleep latency. One approach to this is the use of *adaptive listening* where nodes that lie one or more steps ahead in the path of a transmission can be kept awake for an additional length of time (present as an extension to the basic S-MAC in [24], as well as the T-MAC protocol [26]). This approach provides some reduction in sleep latency at the expense of greater energy expense due to extended activation and overhearing, but is not sufficient for long paths.

### 2.1.2 Energy Efficient Scheduling-based MAC

TRAMA is a schedule-based protocol to provide energy efficient collision free MAC for sensor networks. TRAMA divides time into slots and uses a distributed election scheme based on traffic information at each node to select the transmitter at each time slot. TRAMA reduces energy consumption by reducing collision caused retransmissions and by putting nodes to sleep whenever they do not transmit or receive. To solve the wasted slots problem of traditional TDMA-based MAC, TRAMA allows time slots to be reused by other nodes when the original owner nodes have no traffic to send. Time is organized as signal slots (random-access) and transmission slot (scheduled access).

TRAMA consists of three components: Neighbor Protocol (NP), Schedule Exchange Protocol (SEP) and Adaptive Election Algorithm. NP collects neighbor information by exchanging small signaling packets during signaling slots. Each node periodically sends ”keep-alive” beacons which contains incremental updates about its one-hop neighborhood. A node can then construct the topology of its two-hop neighbors based on the received beacons. SEP establishes and maintains traffic-based schedule information. A node can compute the number of slots in which it has the highest priority among its two hop neighbors. The node then need to announce the intended receiver for these slots, and gives up a slot if it does not have enough packets to send. The schedule is announced via schedule packet in a bitmap structure with each bit corresponds to one particular receiver ordered by their identifiers. The

Adaptive Election Algorithm uses a pseudo-random hash of the concatenation of node's identity and time slot number to calculate its priority. The node with the highest priority in a time slot in the two-hop neighborhood is selected as the sender of the slot. However, if the selected node does not have any packets to send, it can give up the slot which could then be used by another node. Each node with AEA can decide its current state (transmit, receive or sleep) based on priorities from two-hop neighborhood and the schedules from one-hop neighbor.

### 2.1.3 Other MAC protocols

#### 2.1.3.1 IEEE 802.15.4

IEEE 802.15.4 is a new standard to address the need for low-rate low-power low-cost wireless networking. The MAC protocol in IEEE 802.15.4 can operate on both beacon enabled and non-beacon modes. In the beaconless mode, the protocol is essentially a simple CSMA-CA protocol. In beacon mode, the IEEE 802.15.4 uses a superframe structure. A superframe begins with beacon frames sent periodically by the coordinator at an interval that can ranges from $15ms$ to $245s$. There are both active and inactive portion in the superframe. Devices communicate with theirs PANs only during the active period and enter a low power mode during the inactive period. The active portion of each superframe is further divided into 16 equal time slots and consists of three parts: the beacon, a Contention Access Period (CAP) and a Collision Free Period (CFP).

The channel access in the time slots in CAP is contention-based CSMA-CA. In CSMA-CA, a lot of energy is generally consumed by the long backoff period which is required during high traffic periods to avoid collision. IEEE 802.15.4 supports a Battery Life Extension (BLE) mode, in which the CSMA-CA backoff exponent is limited to the range 0-2. This reduces the period of idle listening in low offered traffic applications. A network device can put its radio to sleep to conserve energy immediately after the reception of acknowledgement packet if there is no more data to be sent or received.

The IEEE 802.15.4 standard allows the optional use of CFP for devices that require dedicated bandwidth to achieve low latencies. A device requiring dedicated bandwidth or low-latency transmission can be assigned a *Guaranteed Time Slots* (GTS) in CFP by the PAN coordinator. Each GTS consists of some integer multiple of CFP slots and up to 7 GTS allowed in CFP. When a device wishes to transmit a frame using GTS, it first checks a list on the beacon frame to see whether it has been allocated a valid GTS. If a valid GTS is found, the device enables its receiver at a time prior to the start of the GTS and transmits the data during the GTS period. The MAC of the PAN coordinator ensures that its receiver is enabled for all allocated GTS time slots.

With both CAP and CFP, IEEE 802.15.4 provides a flexible choice to accommodate the needs of different applications and network topologies.

#### 2.1.3.2 ARC

In many sensor network applications, fairness could be a highly desirable metric. For example, to get a whole picture of microorganism environment, roughly the same amount of data from all sensors in the environment is necessary. The authors of [79] proposed an adaptive transmission rate control (ARC) scheme to fairly distribute channel bandwidth between the originating and route-through traffic in the multihop architecture of sensor networks.

The successful injection of a originating packet indicates that there is still capacity available, so the node can increase its data rate. If the injection is failed, it indicates the channel is jammed so the node decreases its data rate. Similar scheme is used for route-through traffic. The fail transmission of a route-through packet will cause the upstream node to decrease its data rate. This implicit signaling can be propagated all the way back to the source node to decrease the amount of route-through traffic.

ARC uses a linear increase and multiplicative decrease approach to adjust the data injection rate of both the originating and route-through traffic. The cost of dropping route-through traffic is higher than originating traffic, so route-through traffic is given preference by having 50% less decreasing rate. Simulations in [79] show that ARC achieves fairness while still maintaining good aggregate bandwidth energy efficiently.

The author also investigate the listening and backoff mechanisms in the CSMA-based MAC. It is recommended that a fixed listening period and exponential decrease backoff period be used to save energy. In the backoff period, a node can be put to sleep instead of idle listening to further conserve energy.

#### 2.1.3.3 Others

The SMACS protocol [77], a schedule-based MAC, achieves network startup and link-layer organization, and the EAR algorithm enables seamless connection of nodes in a sensor network. The neighbor discovery and channel assignment phases are combined to reduce the network setup latency. Network wide synchronization is not needed but communicating neighbors in a subset need to be time-synchronized. SMACS achieves power saving by a random wakeup schedule during the connection phase and by turning the radio off during idle time slots. The authors of [76] proposed a hybrid TDMA/FDMA based centrally controlled MAC scheme. An analytical model is derived to find the

Table 2.1: Overview of MAC protocols for sensor networks

| Protocols | Channel Access | Energy saving techniques | Specific features |
|---|---|---|---|
| PAMAS | Contention | Overhearing avoidance | Turning off radio of nodes nearby the sender and receiver |
| SMAC | Contention | Low duty cycle, overhearing avoidance, message passing | Tradeoff latency and fairness |
| TMAC | Contention | Low duty cycle | Adjust duty cycle |
| IEEE 802.15.4 | Contention & schedule | Low duty cycle | Both contention and contention-free access |
| NAMA | Schedule | None | Distributed election algorithm |
| TRAMA | Schedule | Turn off radio in idle state | Avoid waste time slots |
| ARC | Contention | Fixed listening period | Adaptive rate control to achieve fairness |
| SMACS, EAR | Schedule | Random wakeup and turning radio off when idle | Large available bandwidth compared to sensor data rate |
| Hybrid TDMA/FDMA | Centralized schedule | Hardware based approach | Analysis to find optimal number of channels |
| Multi-channel MAC | Schedule | Turn off radio when idle | Multi-channel assignment, separate low power wakeup radio |
| DMAC | Contention | Low duty cycle | Specific for data gathering tree, Duty cycle adaptation |

optimum number of channels which is most energy efficient. TDMA scheme is favored when the transmitting power is larger while FDMA scheme is favored when receiving power is larger. The Node Activation Multiple Access [74] uses a distributed election algorithm to select only one transmitter per two-hop neighborhood to ensure collision-free receptions for all nodes in the one-hop neighborhood of the transmitter. However, NAMA does not consider energy savings. The authors of [75] proposed a multi-channel MAC, in which each node records the channel used by its one-hop and two-hop neighbors, and make sure its own channel is different from all its two-hop neighbors. Nodes can turn off their data radios to save energy and a separate always on radio is used to wake up nodes that have turned off their main data radio. The wakeup radio consumes much lower power than radios used for regular data communications.

### 2.1.4 Discussion

To have a better understanding of current MAC protocols for sensor networks, we summarize the key features in Table 1. The energy saving techniques show the schemes used in each of these MAC to conserves energy. The specific features shows the novel and important features in each of these MAC protocols.

From the table we can see that the key scheme to be energy efficient is to turn off radio to reduce idle listening. However, this trade off throughput, latency and fairness. Schedule-based MAC is more

energy efficient in nature than contention-based MAC, however with high overhead, thus is worse in terms of self-organization and robustness. Compared to traditional computer networks, sensor networks applications have very different requirements on network topology, traffic pattern, etc. Thus MAC protocols often have very different, sometimes contradictory assumptions. For example, SMAC trades off fairness while [79] chooses fairness as the major goal. Low duty cycle MAC protocols increase latency significantly to save energy, however, latency may be a very important metric in target tracking applications. In terms of traffic pattern, some environment monitoring application need all sensors to report their samples back to the sink while in a fire detection application, traffic are mostly local to enable in-network processing and only one result is required to transmit back to the sink. Thus, we expect that there is not a general MAC protocols suitable for all sensor network applications and specific MAC protocols for a specific application can be simple while have better performance in specific metrics.

Previous sensor network MAC protocols tradeoff other performances, specifically the latency for energy savings. In this thesis, we are interested in designing energy saving MAC without sacrificing latency. In DMAC and DESS, the medium access is contention-based, therefore we use low duty cycle as the energy saving technique. DMAC provides best effort minimum average latency for data gathering application while DESS works to minimize the worst case latency for arbitrary network topology.

## 2.2 Energy efficient Joint Scheduling, Power Control and Routing

Scheduling is another effective approach to save energy. Other energy saving techniques at different layers, such as power control in physical layer and power aware routing in routing layer, can often affect the energy savings. While applying these energy saving techniques separately works, joint optimization approaches would achieve better performance. In this section, we discuss the related works on joint scheduling, power control and routing.

Recently there have been several works ([15], [17], [18], [19], [20]) on jointly scheduling and power control in wireless sensor networks. ElBatt and Ephremides [15, 16] consider the problem of joint scheduling and power control in multi-hop networks. Their solution has two alternating phases: scheduling and power control. A transmission scenario (the selection of a particular set of links to transmit data) is defined as valid if no node is to transmit and receive simultaneously and no node is to receive from more than one neighbor at the same time. An admissible transmission scenario means that a set of transmission power is available to satisfy the $SNR$ constraints for all links in the scenario. In each slot the scheduling algorithm first searches a maximum valid scenario, which then is verified by

the distributed power control algorithm to see if it is admissible. If the valid scenario is not admissible, the scheduling algorithm drops the link with minimum $SNR$ and the power control algorithm is rerun. Once an admissible transmission scenario is found, the sources will transmit data packets using the computed transmission powers in current slot. They also proved that the power control algorithms proposed for cellular network can be applied directly into wireless multi-hop networks. As the scheduling algorithm schedules as many links as possible to be active at each slot, it can not guarantee energy efficiency as discussed in EEJSPC.

The authors in [17] proposed a distributed joint scheduling and power control algorithm for multicasting in wireless Ad Hoc Networks. As in [15], the algorithm in [17] also tries to schedule all links with data transmission requirement. If a set of transmission power can not be found to satisfy the $SNR$ constraints for all the links, the link with Maximum Interference to Minimum Signal Ratio (MIMSR, the ratio between interference and signal strength received at the receiver of the link) is deferred until a feasible power control solution is available. In both [15] and [17], while the power control algorithm is optimal in minimizing the transmission power of a single transmission scenario, the scheduling algorithm which tries to find a maximum valid scenario may result in a non-optimal solution in terms of total energy consumption in multiple slots.

Bhatia and Kodialam [18] derive a performance guaranteed polynomial approximation algorithm for jointly solving routing, scheduling and power control. Given a source and destination, they are interested in making three decisions: the paths the data has to take between the source and the destination, the power with each link transmission is done and the time slots in which specific link transmissions have to take place. However, they consider a different interference model in which the $SINR$ level impacts the average rate rather than the success or loss of individual packets. In our work EEJSPC, as in [15, 17], we will assume an interference model in which a radio can either successfully receive a packet or not depending on the $SINR$ threshold.

A closely related work by Cruz and Santhanam [19] proposes a joint scheduling and power control algorithm to minimize the total average transmission power in the wireless multi-hop network, subject to the constraints on average data rate per link and peak transmission power per node. Similar to [18], they assume an interference model that $SINR$ affects the achieved data rate of the link in a slot. The long-term average rate of a link is then defined as the sum of the achieved data rate per slot divided by number of slots when number of slots goes to infinity. They reduce the problem to a convex optimization problem over a single slot using a duality approach. However, this prior work does not

consider the latency metric explicitly. Although the long-term average rate of a link is guaranteed, if there are latency deadline requirements, many packets could be useless even if they reach the sink.

The authors in [67] consider the problem of power controlled minimum frame length scheduling for TDMA wireless networks. Given a set of one-hop transmission requests, their objective is to schedule the transmission requests in a minimum number of time slots. The consider per-slot and per-frame versions of the problem and develop mixed integer linear programming models. To minimize the frame length, their approach is to schedule the maximal feasible active links in each slots, same as [15, 16]. Thus energy efficiency can not be guaranteed.

Sichitiu [59] proposes a cross-layer scheduling for power efficiency in wireless sensor networks. In order to conserve energy, sensor nodes are turned off. However, since an in active sensor node is no longer part of the network, the network can become disconnected. The author proposes a deterministic, schedule-based energy conservation scheme, in which time-synchronized sensors form on-off schedules that enable the sensor to be awake only when necessary. The scheme can be decoupled into two distinct phases for each flow in the network: the setup and reconfiguration phase, and the steady state phase. In the setup and reconfiguration phase, first a route from the node originating the flow to the base station is selected, then the schedules are set up along the chosen route. If a schedule can not be set up along the chosen route, the routing protocol will find an alternative route. In this scheme, the scheduling and routing schemes work separately.

Previous related works mainly focused on the metrics of energy efficiency and did not explicitly consider the latency. In our work EEJSPC, we consider the energy efficiency of joint scheduling and power control under a specific latency bound. And in MLSR, we are interested in minimum latency joint scheduling and routing. In both works, we explicitly take latency into the design consideration. Our approaches can achieve better tradeoffs between energy and latency than previous works.

# Chapter 3

# DMAC: Tree-based Data Gathering MAC

## 3.1 Overview

In the Introduction Chapter 1, we described briefly the fundamental tradeoffs between energy efficiency and low latency. In this chapter, we will describe the tradeoff in detail. And in a typical data gathering application scenario, we employ the unique feature of the unidirectional communication pattern to propose an adaptive MAC for tree-based data gathering which can provide bother energy efficiency and low latency.

We know that in order to save energy, we need to turn off the radio to avoid energy waste of idle listening. Since in sensor network applications, traffic load is very light most of the time, it is often desirable to turn off the radio when a node does not participate in any data delivery. Prior work, e.g. [63] suggests putting idle nodes in power saving mode and switching nodes to full active mode when a communication event happens. However, even when there is traffic, idle listening still may consume most of the energy. For example, a sensor node reports its sensing reading one packet per second. Suppose the packet length is 100 byte, it takes $8ms$ for a radio of 100Kbps data rate, while the other $992ms$ is still wasted in idle listening. S-MAC [60] reduces idle listening energy cost by reducing the duty cycle of a sensor node in which a node follows a periodical active/sleep schedule. During sleep period, nodes turn off radio to preserve energy. During active period, nodes turn on radio to Tx/Rx messages.

Although a low duty cycle MAC is energy efficient, it has three side-effects. First, it increases the packet delivery latency. At a source node, a sampling reading may occur during the sleep period and has to be queued until the active period. An intermediate node may have to wait until the receiver wakes up before it can forward a packet received from its previous hop to the next hop. This is called *sleep latency* in SMAC [60], and it increases proportionally with hop length by a slope of schedule length

(active period plus sleep period). Secondly, a fixed duty cycle does not adapt to the varying traffic rate in sensor network. A fixed duty cycle for the highest traffic load results in significant energy wastage when traffic is low while a duty cycle for low traffic load results in low message data delivery and long queuing delay. Therefore it is desirable to adapt the duty cycle under variant traffic load. Thirdly, a fixed synchronous duty cycle may increase the possibility of collision. If neighboring nodes turn to active state at the same time, all may contend for the channel, making a collision very likely.

There are several works on reducing sleep delay and adjusting duty cycle to the traffic load. Those mechanisms are either implicit (e.g. [60], [26]) in which nodes remain active on overhearing of ongoing transmission or explicit (e.g. [62]) in which there are direct duty cycle adjusting messages. SMAC [60] proposed adaptive listening to reduce the sleep delay. In adaptive listening, a node who overhears its neighbor's transmission wakes up for a short period of time at the end of the transmission. In this way, if the node is the next-hop node, its neighbor is able to immediately pass the data to it instead of waiting for its scheduled listen time. An ongoing work to improve SMAC [30] points out that the phase difference in the schedule could affect the latency. It includes a simple analysis for two cases. In case 1 where the phase difference is in the same direction of the data flow, delay is reduced. In case 2 where phase difference is in the opposite direction, delay is increased. Then it proposes a scheme to design global synchronization algorithm.

In TMAC [26], a node keeps listening and potentially transmitting as long as it is in active period. An active period ends when no activation event has occurred for a certain time. The activation time events include reception of any data, the sensing of communication on the radio, the end-of-transmission of a node's own data packet or acknowledgement, etc. FRTS is employed to solve the early sleep problem. The authors of [62] propose a slot-based power management mechanism. If the number of buffered packets for an intended receiver exceeds a threshold $L$, the sender signals the receiver to remain on for the next slot. A node requested to stay awake sends an acknowledgement to the sender, indicating its willingness to remain awake in the next slot. The sender can then send a packet to the receiver in the following slot. The request is renewed on a slot-by-slot basis.

However, in previous implicit or explicit mechanisms, not all nodes beyond one hop away from the receiver can overhear the data communication, and therefore packet forwarding will stop after a few hops. As we shall describe in section 3.2, this *data forwarding interruption problem* causes sleep latency for packet delivery.

After describing the data forwarding interruption problem, we will describe the proposed DMAC mechanism in section 3.3. DMAC employs a *staggered active/sleep schedule* to solve this problem and

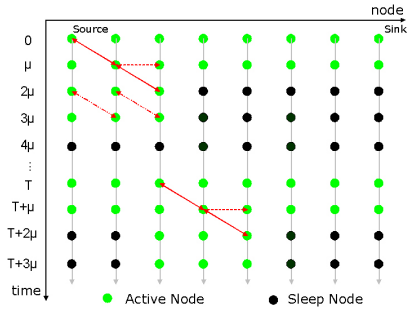



Figure 3.1: SMAC with adaptive listening in a chain.

enable continuous data forwarding on the multihop path. In DMAC, *data prediction* is to used enable active slot request when multiple children of a node have packets to send in a same sending slot, while *More to Send packet* is used when nodes on the same level of the data gathering tree with different parents compete for channel access. In section 3.4 and 3.4.4, we evaluate the performance of DMAC by simulation and real mote experiments.

## 3.2 Data Forwarding Interruption Problem

The data forwarding interruption problem exists in implicit adaptive duty-cycle techniques because the overhearing range is limited by the radio's sensitivity to signals on air. Nodes that are out of the hearing range of both the sender and the receiver are unaware of ongoing data transmissions, and therefore go to sleep until the next cycle/interval. The data forwarding process will then stop at the node whose next hop towards the sink is out of the overhearing range because it is in sleep mode. Packets will then have to be queued until the next active period which increases latency. Also, for explicit mechanism, the duty cycle adjusting messages can only be forwarded limited hops in an active period. So nodes out of the range go to sleep after their basic duty cycle, leading to interrupted data forwarding.

Assume an active period (i.e. the portion of time in each interval when a node is active, unless there is more data to be sent/received) is only long enough to transmit one packet each hop. In SMAC, only the next hop of the receiver can overhear the data transmission and remains active for a long period. Other nodes on the multihop path do not overhear the data transmission thus go to sleep after the basic active period, resulting in the interruption of packet forwarding to the sink till the next duty cycle. It is shown theoretically in [60] that the delay with adaptive listening still increases linearly with the number of hops with a slope that is half of the interval length. Therefore, compared with the case of no adaptive listening, the delay is only reduced by half. Meanwhile, nodes other than the next-hop in the

neighborhood of the sender and the receiver also overhear the data transmission and thus may remain active unnecessarily. Similarly, in TMAC [26], a node remains active if it senses any communication on the air. Typically, a radio's interference range is larger than its transmission range (e.g. in NS-2, the interference range is set to more than twice the transmission range). In TMAC, any neighbor nodes in the interference range of either the sender or the receiver will remain active. Many of the nodes do not participate in the data delivery but remain active for an unnecessarily long period which wastes energy. Meanwhile only nodes in the interference range hear the communication, while other nodes out of the interference range on the multi-hop path still go to sleep after their basic active period. Thus packets still suffer from the data forwarding interruption problem. The use of the FRTS technique proposed in TMAC can only help forward the packet one hop further. Besides the sleep latency, the duty adjustment also suffers from this early sleep problem. The same problem happens to the technique in [62], in which the request for a next active slot can be only received by the next hop. The nodes beyond that will still go to sleep after their basic active period.

Figures 3.1 illustrates this data forwarding interruption problem using SMAC with adaptive listening as an example. There is a chain of nodes with a single source on the far left and the sink on the far right. We assume an active period is only long enough to transmit one packet one hop. By adaptive listening, the next hop of the receiver overhears the receiver's ACK or CTS packet, then remain active an additional slot. But other nodes still go to sleep after their active periods. If the source have multiple packets to send, those packets can only be forwarded two hops away every interval $T$. Latency is also only reduced by half. Collision is also depicted in the figure. Suppose in slot between $2\mu$ and $3\mu$, both node 0 and node 1 need to transmit packets, a collision could happen. Things will be even worse if between 0 and $\mu$, all nodes have packets to send.

The hearing/interference range also causes a tradeoff between the latency and energy. If the hearing range is long, latency is reduced since more nodes on the path can overhear the communication and remain active. Meanwhile, more nodes not on the path also overhear the communication and waste energy in idle listening on the increased active periods. We need a MAC that can tell all nodes on the path and no other nearby nodes to stay active and/or increase their duty cycles to enable continuous data forwarding without incurring energy waste of unrelated nodes.

## 3.3 DMAC Protocol Design

### 3.3.1 Staggered Wakeup Schedule

One can identify three main communication patterns in sensor network applications. The first involves local data exchange and aggregation purely among nearby nodes (these can be handled by clustering or simple medium access mechanisms). The second involves the dispatch of control packets and interest packets from the sink to sensor nodes. Such sink-originated traffic is small in number and may not be latency sensitive. We can reserve a separate active slot periodically with a larger interval length for such control packets. The third and most significant traffic pattern in WSN is data gathering from sensor nodes to sink. For a sensor network application with multiple sources and one sink, the data delivery paths from sources to sink are in a tree structure, an *data gathering tree* [32]. Routes may change during data delivery, but we assume that sensor nodes are fixed without mobility and that a route to the sink is fairly durable, so that a data gathering tree remains stable for a reasonable length of time. Flows in the data gathering tree are unidirectional from sensor nodes to sink. There is only one destination, the sink. All nodes except the sink will forward any packets they received to the next hop (except local processing packets which are handled in cluster). Our key insight in designing a MAC for such a tree is that it is feasible to stagger the wakeup scheme so that packets flow continuously from sensor nodes to the sink. *DMAC is proposed to deliver data along the data gathering tree*, aiming at both energy efficiency and low latency.

In DMAC, we stagger the activity schedule of nodes on the multihop path to wake up sequentially like a chain reaction. Figure 3.2 shows a data gathering tree and the staggered wakeup scheme. An interval is divided into receiving, sending and sleep periods. In receiving state, a node is expected to receive a packet and send an ACK packet back to the sender. In sending state, a node will try to send a packet to its next hop and receive an ack packet. In sleep state, nodes will turn off radio to save energy. The receiving and sending period have same length of $\mu$ which is enough for one packet transmission and reception. Depending on its depth $d$ in the data gathering tree, a node skews its wakeup scheme $d\mu$ ahead from the schedule of the sink. In this structure, data delivery can only be done in one direction towards the root. Intermediate nodes have a sending slot immediately after the receiving slot.

A staggered wake-up schedule has four advantages. First since nodes on the path wake up sequentially to forward a packet to next hop, sleep delay is eliminated if there is lost due to channel error or collision. Second, a request for longer active period can be propagated all the way down to the sink, so that all nodes on the multihop path can increase their duty cycle promptly to avoid data stuck in

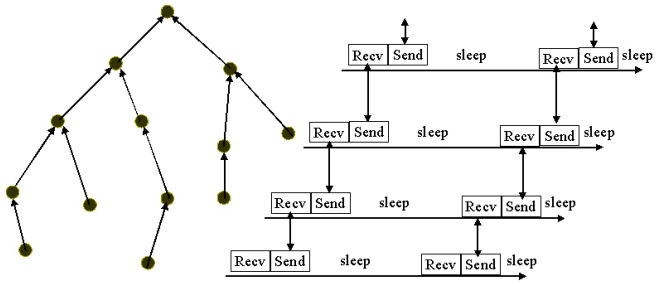



Figure 3.2: DMAC in a data gathering tree.

intermediate nodes. Third, since the active periods are now separated, contention is reduced. Fourth, only nodes on the multihop path need to increase their duty cycle, while the other nodes can still operate on the basic low duty cycle to save energy.

In a multi-hop wireless network, it is well known that contention-based MACs suffer from the hidden node problem. In MACAW [34], virtual and physical carrier sense and RTS/CTS exchange are utilized to reduce hidden node problem. For large packet sizes, these small control packets are efficient in saving the possible high cost of a packet lost. However, for sensor networks where packet size is usually small, the overhead of RTS/CTS could be very high compare to the actual data transmission cost. Therefore we do not advocate the use of RTS/CTS in DMAC. DMAC, however, employs link layer ARQ through ACK control packet and data retransmission, and the hidden node problem is mitigated to some extent through the manner in which active slots are scheduled so that nodes on the same path do not cause hidden node collisions. Although ACK packets consume energy and bandwidth, we believe these are essential for the link reliability to recover lost packet due to harsh quality wireless channel and contention (though there is always the possibility of using implicit ACKs [61] in case of highly reliable links). If a sending node does not receive an ACK packet from receiving node, it will queue the packet until next sending slot. After 3 retransmission, the packet will be dropped.

In DMAC, nodes with the same depth will have same offset, and thus a synchronous schedule. During the sending period, nodes will compete for the channel. To reduce collision during this period, every node backs off for $difs$ plus a random time within a fixed contention window at the beginning of a sending slot. Since the length of a sending slot is only enough for one packet transmission, there is no need for exponential contention window increase, and therefore we employ a fixed contention window.

Based on the above choices, the sending and receiving slot length $\mu$ is set to:

$$\mu = DIFS + CW + DATA + sifs + ACK$$

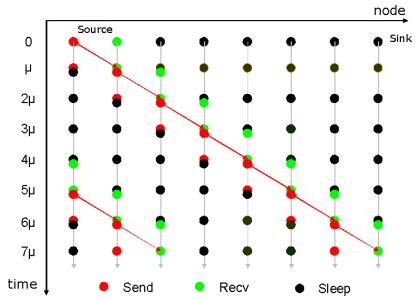



Figure 3.3: DMAC in a chain.

where DIFS is the DCF inter-frame space, CW is the fixed contention window size, $DATA$ is the packet transmission time(We assume all packets are in the same length), $sifs$ is Short inter-frame space and $ACK$ is the ACK packet transmission time.

Synchronization is needed in DMAC. However, local synchronization is enough since a node only needs to be aware of its neighbors' schedule. There exist techniques such the reference broadcast synchronization scheme (RBS)[27] that can achieve time synchronization precision of $3.68 \pm 2.57 \mu sec$ after 4 hops. Given that typical slot lengths are on the order of $10ms$ in length, we will assume that synchronization is available in the following discussions.

### 3.3.2 Data Delivery and Duty Cycle Adaption in Multihop chain

Figure 3.3 shows DMAC operation in a multihop chain. Every node periodically turns to receiving, sending and sleep states. It is shown that when there is no collision, a packet will be forwarded sequentially along the path to the sink, without sleep latency.

However when a node has multiple packets to send at a sending slot, it needs to increase its own duty cycle and requests other nodes on the multihop path to increase their duty cycles too. We employed a slot-by-slot renewal mechanism. We piggyback a *more data* flag in the MAC header to indicate the request for an additional active periods. The overhead for this is very small. Before a node in its sending state transmits a packet , it will set the packet's *more data* flag if either its buffer is not empty or it received a packet from previous hop with *more data* flag set. The receiver check the *more data* flag of the packet it received, and if the flag is set, it also sets the *more data* flag of its ACK packet to the sender. With the slot-by-slot mechanism and the policy to set *more data* flag when buffer is not empty, DMAC can react quickly to traffic rate variation to be both energy efficient and maintain low data delivery latency. A node will decide to hold an additional active period if:

1. It sends a packet with the *more data* flag set and receives an ACK packet with the *more data* flag set.

2. It receives a packet with *more data* flag set.

In DMAC, even if a node decides to hold an additional active period, it does not remain active for the next slot but schedules a $3\mu$ sleep then goes to the receiving state as shown in Figure 3.3. The reason for a $3\mu$ sleep is that it knows the following nodes on the multihop path will forward the path in the next 3 slots. In [25], it is shown that the maximum utilization of a chain of ad hoc nodes is $\frac{1}{4}$ if the radio's interference range is twice the transmission range. So the maximum sending rate for a node is one packet per 4 slots. However, to accommodate the possibility of short range between two neighbor nodes, a node will only send one packet every $5\mu$ in DMAC in order to avoid collision as much as possible. Of course, this may reduce the maximum network capacity by about 20%, but if the traffic load is more than 80% of the maximum channel capacity duty-cycled mechanisms would not function efficiently in any case, making this a moot point.

A good result of the staggered wake up schedule is that the *more data* flag can be propagated to all the nodes on the multi-hop path. In Figure 3.3, suppose the source sets the *more data* flag of the first packet, since this packet can be forwarded to the sink without interruption, all nodes will receive the first packet with *more data* flag set thus will hold an additional active period $3\mu$ later after their sending slot. So at time $5\mu$, the second packet from the source can still be delivered to the sink with very short delay.

However, there is a possibility of inconsistency on the new active period request. We may have a situation where the receiving node is awake, while the sending node is off. This could happen when the receiving node received a packet with *more data* flag, but the ACK packet sent by the receiver is not received by the sender. In this case, the receiving node will waste an active period in idle listening. However, the slot-by-slow renewal mechanism will make sure that a node will only waste one additional active period, though packets will have a sleep delay. The situation where the sending node is awake but the receiving node is off is not possible since the sending node will hold an additional active period only if it successfully received an ACK packet with *more data* which guaranteed the receiver is awake. DMAC avoids this situation because transmission is more energy costly than receiving and a packet retransmission chance will be wasted.

Measurements have showed that the cost for switching radio between active and sleep is not free. However, the overhead of this switching is likely to be small [29] compared to energy savings in a $3\mu$ sleep period of around $30ms$.

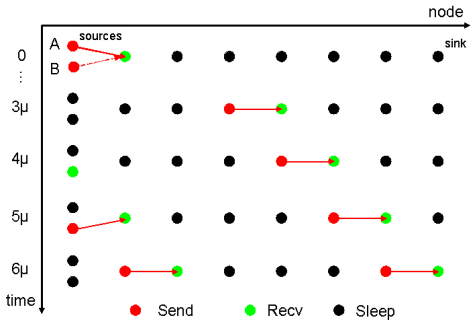



Figure 3.4: Data prediction scheme reduces sleep delay.

### 3.3.3 Data Prediction

In last section, we assume a single source needs a higher duty cycle than the basic lower duty cycle. In a data gathering tree, however, there is a chance that each source's rate is small enough for the basic duty cycle, but the aggregated rate at an intermediate node exceeds the capacity of basic duty cycle. For example, suppose a node C has 2 children A and B. Both children has only one packet to send every interval. At the sending slot of an interval, only one child can win the channel and send a packet to the node. Assume A wins the channel and sends a packet to C. Since A's buffer is empty, the *more data* flag is not set in A's packet. C then goes to sleep after its sending slot without a new active period. B's packet would then have to be queued until next interval. This results in sleep delay for packets from B.

We propose a scheme called *data prediction* to solve this problem. If a node in receiving state receives a packet, it predicts that its children still have packets waiting for transmission. It then sleeps only $3\mu$ after its sending slot and switches back to receiving state. All following nodes on the path also receive this packet, and schedule an additional receiving slot. In this additional data prediction receiving slot, if no packet is received, the node will go to sleep directly without a sending slot. If a packet is received during this receiving slot, the node will wake up again $3\mu$ later after this sending slot.

For a node in sending state, if during its backoff period it overhears the ACK packet from its parent in the data gathering tree, it knows that this sending slot is already taken by its brother but its parent will hold an additional receiving slot $3\mu$ later, so it will also wake up $3\mu$ later after its sending slot. In this additional sending slot, the node then can transmit a packet to its parent. Figure 3.4 shows an example of the *data prediction* scheme.

Of course, this generalizes beyond the case of a node having two children. If a node has more children, in the additional receiving slot, the remaining children would compete for the channel again. This process would repeat until eventually, all children will be able to transmit their packet to the parent one by one with shortest delay. However if a collision happens, all children nodes have to wait until

next interval. But since those nodes have the same parent, they are at most two hops away. Hence they can detect each other's transmission, and the chance of a collision due to hidden node problem is small.

There is an overhead brought by the *data prediction* scheme. After the reception of the last packets from its children, a node will remain idle for a receiving slot which waste energy in idle listening. Compared to the huge latency reduction by the *data prediction*, we believe this additional overhead would be worthwhile.

### 3.3.4 MTS

Although a node will sleep $3\mu$ before an additional active period to avoid collision, there is still a chance of interference between nodes on different branches of the tree. Consider the example in Figure 3.5; two nodes A and B are in interference range of each other but have different parents in the data gathering tree. In the sending slot of one interval, A wins the channel and transmits a packet to its parent. Neither B nor its parent C holds additional active slots in this interval. Thus B can only send its packet in the sending slot of next interval, resulting a sleep latency of $T$. Since C does not receive any packet in its receiving slot and B does not overhear ACK packet from C in its sending slot, *data prediction* scheme will not work.

We propose a solution to mitigate this interference using an explicit control packet, that we refer to as *More to Send* (MTS). The MTS packet is very short with only destination's local ID and a flag. A MTS packet with flag set to 1 is called a request MTS. A MTS packet with flag set to 0 is called a clear MTS.

A node sends a request MTS to its parent if either of the two conditions is true:

1. It cannot send a packet because of channel busy. After the node's backoff timer fires, it finds there is not enough time for it to send a packet and it does not overhear its parent's ACK packet. It then assumes that it lost the channel because of interference from other nodes.

2. It receives a request MTS from its children. This is aimed to propagate the request MTS to all nodes on the path.

A request MTS is sent only once before a clear MTS packet is sent.

A node sends clear MTS to its parent if the following three conditions are true:

1. Its buffer is empty.

2. All request MTSs received from children are cleared.

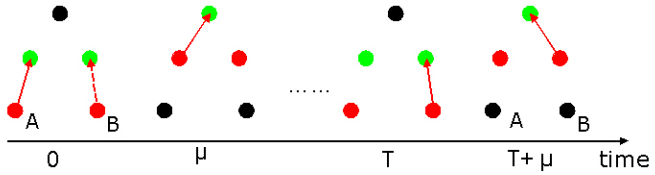



Figure 3.5: Interfence between two sending nodes causes sleep delay.

Table 3.1: Radio parameters

| | |
|---|---|
| Radio bandwidth | 100Kbps |
| Radio Transmission Range | 250 m |
| Radio Interference Range | 550 m |
| Packet Length | 100 bytes |
| Transmission Power | 0.66W |
| Reception Power | 0.395W |
| Idle Power | 0.35W |

3. It sent a request MTS to its parent before and has not sent a clear MTS.

A node which sent or received a request MTS will keep waking up periodically every $3\mu$. It switches back to the basic duty cycle only after it sent a clear MTS to its parent or all previous received request MTS from its children were cleared.

Same as the slot-by-slot renewal scheme and data prediction scheme, the higher duty cycle request by MTS packets are forwarded through the staggered schedule to all nodes on the multihop path. The difference from the slot-by-slot renewal scheme is that only two MTS packets are sent for a MTS request/clear period. This is due to the overhead of the MTS packets. If a MTS packet need to be transmitted in each additional active period, the overhead of MTS packets will be high.

Inconsistent schedule is possible due to the loss of MTS packets. A soft timer is maintained to clear request MTS if no data received or transmitted after a certain number of receiving slot in order to avoid unnecessary active slots because of lost of clear MTS packets.

Slot length has to be increased to enable the transmission of MTS packets after a data transmission. Since the MTS packet is very short, the increase will be very small. Energy consumption will increase too because the overhead of MTS packets and the increase of slot length. In the simulation section, we show that MTS can significantly reduce latency in a sensor network at only small overhead of energy cost.

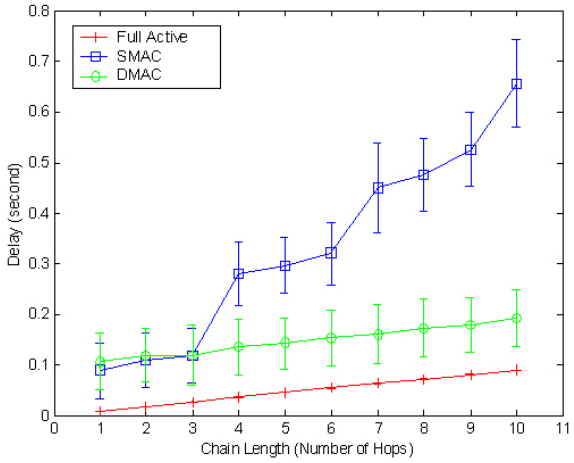



Figure 3.6: Mean packet latency on each hop under low traffic load.

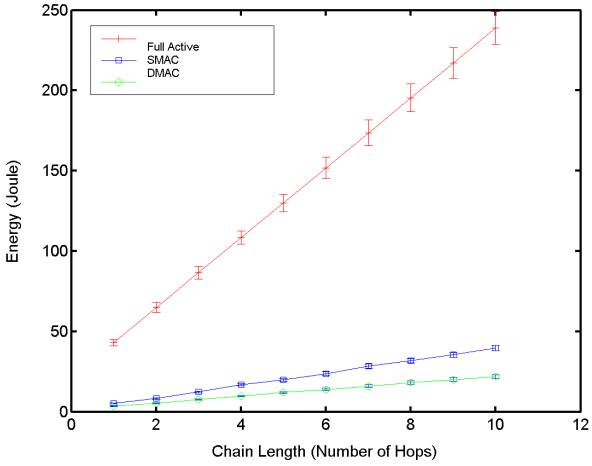



Figure 3.7: Total energy consumption on each hop under low traffic load.

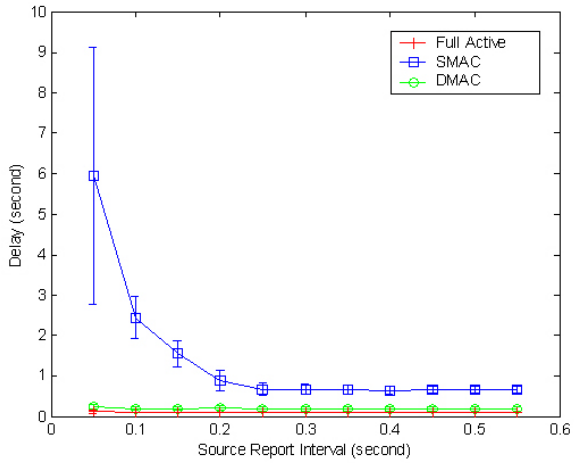



Figure 3.8: Mean packet latency for 10 hops chain under different source report interval.

## 3.4 Performance Evaluation

We implemented our prototype in the ns-2 network simulator with the CMU wireless extension. For comparison, we also implement a simple version of SMAC with adaptive listening, but without its synchronization and message passing scheme. We will also compare with a full active CSMA/CA MAC without periodical sleep schedule. This will serve as the baseline of latency, energy and throughput performance.

We choose 3 metrics to evaluate the performance of DMAC: **Energy Cost** is the total energy cost to deliver a certain number of packets from sources to sink. This metric shows the energy efficiency of the MAC protocols. **Latency** is the end to end delay of a packet. **Throughput or Delivery ratio** is the ratio of the number of packets arrived at the sink to the number of packet sent by sources.

The radio characteristics are shown in Table 3.1. The energy costs of the Tx:Rx:Idle radio modes is set to 1.67:1:0.88. The sleeping power consumption is set to 0. A MTS packet is 3 bytes long.

According to the parameters of the radio and packet length, the receiving and sending slot $\mu$ is set to $10ms$ for DMAC and $11ms$ for DMAC/MTS. The active period is set to $10ms$ for SMAC with adaptive listening. All schemes have the basic duty cycle of 10%. This means a sleep period of $180ms$ for DMAC, $198ms$ for DMAC/MTS and $90ms$ for SMAC.

All simulations are run independently under 5 different seeds. All sources generate packets at constant averaged rate with randomization in inter-packet interval.

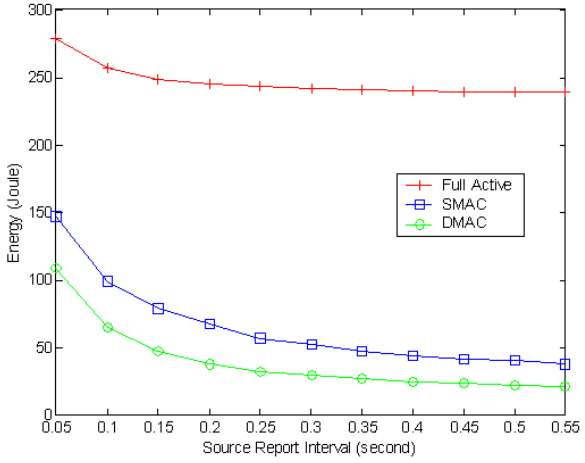



Figure 3.9: Energy consumption for 10 hops chain under different source report interval.

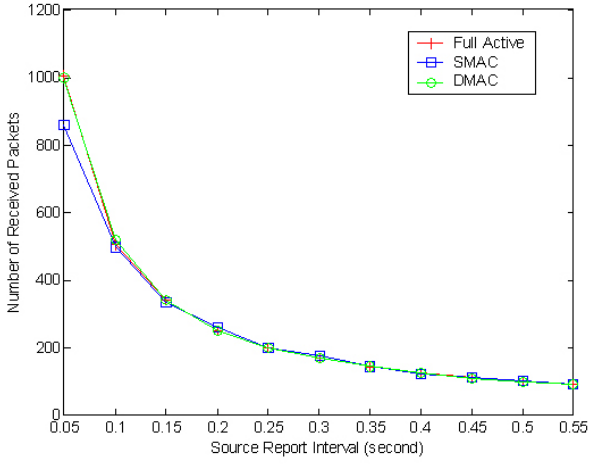



Figure 3.10: Throughput for 10 hops chain under different source report interval.

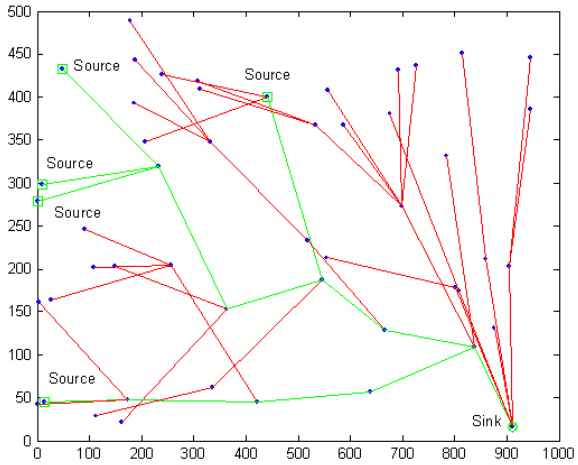



Figure 3.11: A random data gathering tree.

### 3.4.1 Multihop chain

To reveal the fundamental performance of DMAC, we first perform a test on a simple multihop chain topology with 11 nodes. The distance between adjacent nodes is 200 meters. First in order to show the capability of reducing the sleep delay in DMAC, we measure the end-to-end latency of packets under very light traffic rate of source report interval $0.5s$. In this light traffic load, there is no queuing delay but only a sleep delay that is caused by periodical sleep.

Figure 3.6 shows the averaged packet latency with different hop length. In both DMAC and full active CSMA/CA, the latency increase linearly with the number of hops with almost the same slop. The additional latency of DMAC is at the source when a sensing reading occurs during the sleep period and has to wait until the node wakes up. The SMAC with adaptive listening, however has higher latency. Especially, the latency has a jump every 3 hops. This is because by adaptive listening, a packet can be forwarded two hops instead of one hop without adaptive listening. However the packet has to queued for a schedule interval for the third hop. This is shown clearly in the figure.

Figure 3.7 shows the energy cost with different hop length. In all MAC protocol, the energy cost increase linearly with the number of hops. However, the energy cost of the full active CSMA/CA increases much faster than other two MAC protocols. DMAC consumes less energy cost than SMAC. This is due to the additional active period in SMAC for nodes that are not the next hop of a data packet (but are within hearing range).

We then test the rate adaption of these MAC protocols. We vary the traffic load by changing the sensor report interval on the source node from 0.05s to 0.55s. The hop length is fixed at 10 hops.

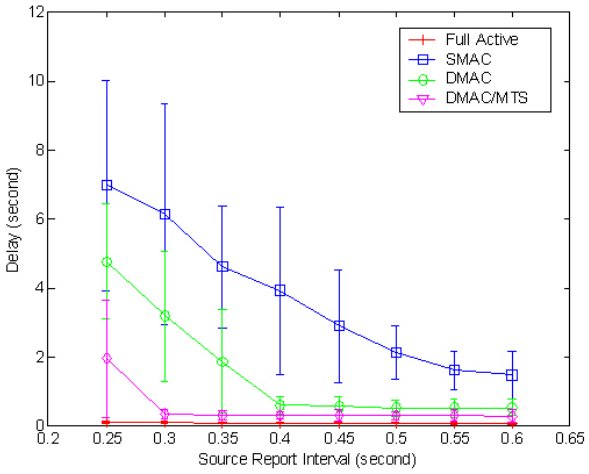



Figure 3.12: Mean packet latency for a data gathering tree under different traffic load.

Figure 3.8 shows the averaged packet latency under different source report interval. Clearly full active CSMA/CA has the lowest latency. DMAC has a slightly higher latency due to the initial latency at the source. SMAC, however, has much higher latency, especially when traffic load is heavy at small source report interval. The reason is that since packets can be forwarded two hops every one interval, those packets suffered from both sleep delay and queuing delay. When traffic load is very high, collision would significantly increase packet latency as a retransmission can only be done after one total schedule interval. When source report is less than 0.05s, the traffic load will be more than 80% of the maximum channel capacity. Only full active CSMA/CA can handle such a high traffic load.

Figure 3.9 shows the total energy cost under different source report interval. Energy cost decreases as traffic load decreases. For full active CSMA/CA, however, the decrease is small since without radio off, the idle listening still consume significant energy. DMAC has a less energy cost due to the same reason as above that nodes other than next hop of a data packet remain active unnecessarily.

Figure 3.10 shows the throughput achieved for different MAC protocols. All MAC schemes have quite good data delivery ratio near 1 under the simple multihop chain topology.

### 3.4.2 Random Data gathering Tree

In this topology, 50 nodes are distributed randomly in a $1000m \times 500m$ areas shown in Figure 3.11. The sink node is at the right bottom corner. A data gathering tree is constructed by each node choosing from its neighbor the node closest to the sink as its next hop. In order to show the different packet latency, a

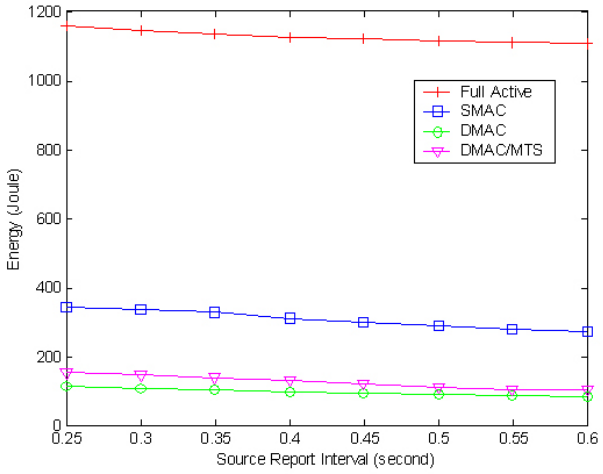



Figure 3.13: Energy consumption for a data gathering tree under different traffic load.

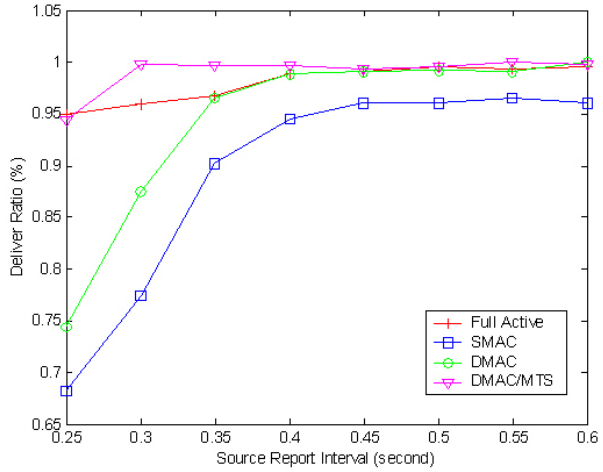



Figure 3.14: Data delivery ratio for a data gathering tree under different traffic load.

source should be at least 3 hops away from the sink. Five margin nodes are chosen as sources to testify the mechanism of data prediction and MTS. All sources generate reports at the same rate.

The packet latency under different source report intervals is shown in Figures 3.12. Full active CSMA/CA has small delay for all traffic load. However, other three MAC schemes' latency increases significantly when the traffic load is larger than a certain threshold. DMAC/MTS can handle the highest traffic load with small delay among the three MAC schemes with periodical sleep. Compared to mulithop chain under the same heavy traffic load, the latency in a data gathering tree is much higher. This is due to the interference between nodes in the same depth of the tree. The interference could result in data lost, schedule inconsistency and MTS packet lost which increase the sleep latency.

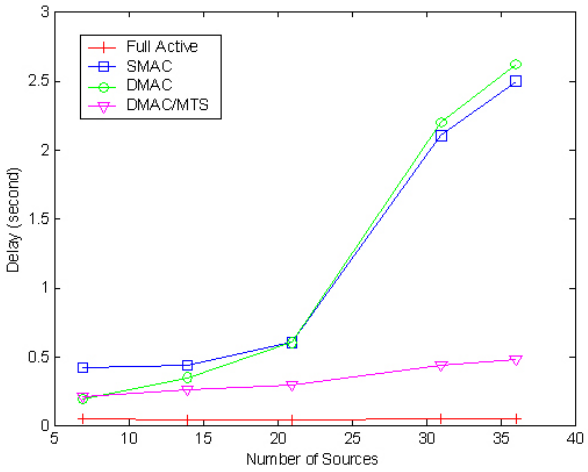



Figure 3.15: Mean packet latency for data gathering different source number.

Figure 3.13, 3.14 shows the energy and throughput performance. We collect the energy costs of all the 50 nodes in the network because some MAC schemes could cause unrelated nodes to maintain higher duty cycle. It is shown in the figure that DMAC and DMAC/MTS are the two most energy efficient MAC schemes. DMAC/MTS, however, consumes higher energy than DMAC because of the overhead of MTS packets and more active period requested by MTS packets. In terms of end-to-end throughput, DMAC/MTS has a good delivery ratio while SMAC and DMAC's delivery ratio decreases when traffic load is heavy.

We further evaluate the scalablity of DMAC under a dense network, in which 100 nodes are randomly placed in a $100m \times 500m$ area. A data gathering tree is constructed rooted at the sink on the right bottom corner. All sources generate traffic at one message per 3 seconds. We vary the number of sources which are chosen randomly from the margin nodes in the network.

Figure 3.15 shows the averaged delay under different number of sources. As source number increases, inteference increases which results in increased latency for SMAC and DMAC without MTS. DMAC/MTS, however, can still maintain quite low latency. This low latency is achived at very small overhead in energy compared to DMAC without MTS, which is shown in figure 3.16. DMAC/MTS also has the second delivery ratio next to full active CSMA. This clearly shows the effectness of DMAC/MTS.

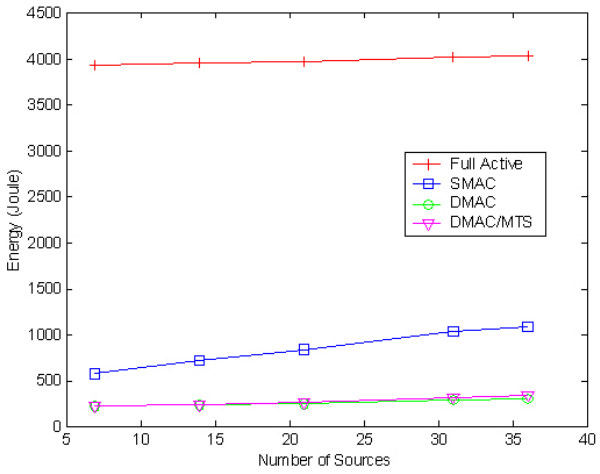



Figure 3.16: Energy consumption for data gathering different source number.

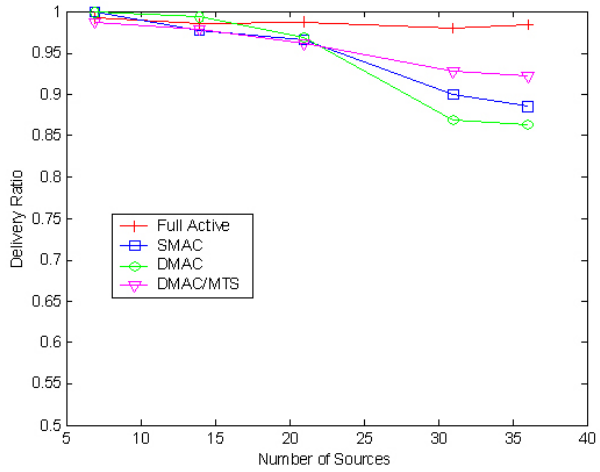



Figure 3.17: Data delivery ratio for data gathering with different source number.

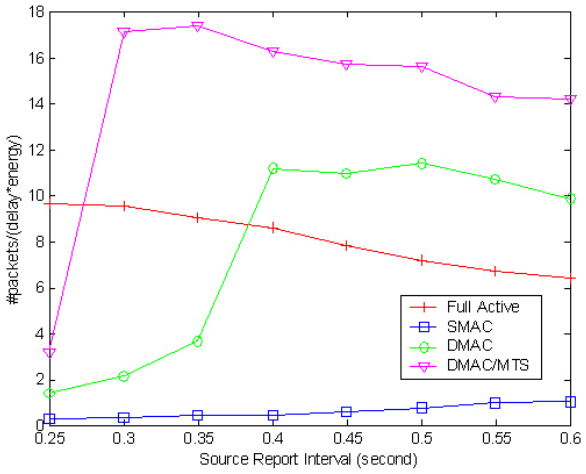



Figure 3.18: Trade off among energy, latency and throughput for a data gathering tree under different traffic load.

### 3.4.3 Energy-Throughput-Latency Tradeoffs

To understand the tradeoffs among energy, throughput and latency, Figure 3.18 shows the number of packets can be sent per unit resource measured in terms of $Energy \times Latency$ for scenario in Figure 3.11. From the figure, we see that because SMAC achieves energy efficiency at the sacrifice of latency, it sends the least number of packets per $Joule \times Second$. This suggests that for applications that can tolerate message latency, SMAC is a reasonable solution. But for applications that require real-time data delivery, SMAC is not feasible due to the data forwarding interruption problem. DMAC and DMAC/MTS, however, can achieve both energy efficiency and low message latency. DMAC/MTS can operate with even smaller base duty cycle to save more energy when traffic is light and can still adapt to traffic bursts with high throughput, low latency and small energy consumption. However, this figure also shows that when traffic load exceeds a certain threshold, a full active MAC is most suitable when taking both energy and delay into account.

Since DMAC can adjust duty cycle to traffic load with small latency, we can set the basic duty cycle even smaller. But a lower duty cycle could have longer initial sleep delay at the source node when a sensing reading occurs during the source's radio is off. So there is a limitation on lowest basic duty cycle DMAC can operate on. However, with the same application latency bound requirement, DMAC can operate on a lower basic duty cycle than SMAC or TMAC to be more energy efficient.

Finally, we should note that this comparison between DMAC and SMAC is only applicable under the specific data gathering tree scenario for unidirectional communication flow from multiple sources to a single sink. SMAC is in fact a general-purpose energy-efficient MAC that can handle simultaneous

Table 3.2: MICA2 Radio parameters

| | |
|---|---|
| Radio bandwidth | 19.2Kbps |
| Packet Length | 36 bytes |
| Transmission Power | 25mW |
| Reception Power | 28mW |
| Idle Power | 18mW |

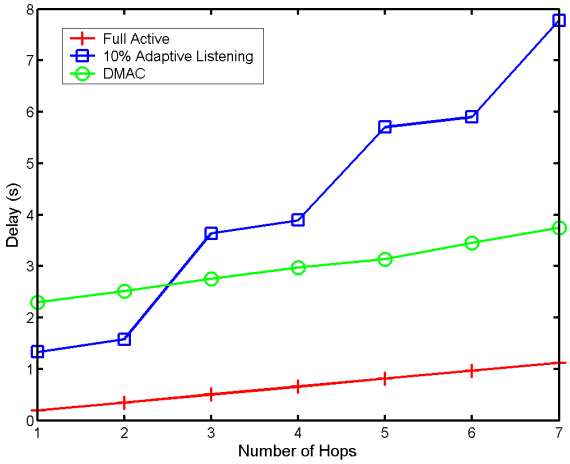



Figure 3.19: Mean packet latency on each hop under low traffic load in Mote experiments.

data transmissions and flows between arbitrary source and destination. For applications that require data exchange between arbitrary sensor nodes, DMAC cannot be used while SMAC will be a good choice.

### 3.4.4 Experimental Results

To further evaluate DMAC performance on a real system, we have implemented DMAC on MICA2 Mote [35]. The basic radio features of the MICA2 Mote is shown in table 3.2. We have thus far tested DMAC on the multi-hop chain topology to reveal its fundamental performance and validate the corresponding simulation results 1. The distance between two adjacent nodes is 0.6m. We configured the MICA2 radio [3] to transmit using the smallest transmission power (output power .20dBm) so that a node can only reach its direct one hop neighbor nodes. In DMAC, the receive and transmit slot length is set at 200ms, the sleep period is 3600ms, so the total duty cycle is 10%. To have a fair comparison, the active slot length of SMAC is also set at 200ms but the sleep length is only 1800ms to have 10% duty cycle.

First, in order to only show the reduced sleep latency of DMAC, we measure the end-to-end latency of each packet under a very light traffic setting of one packet per 12 seconds. Each packet is 36 bytes

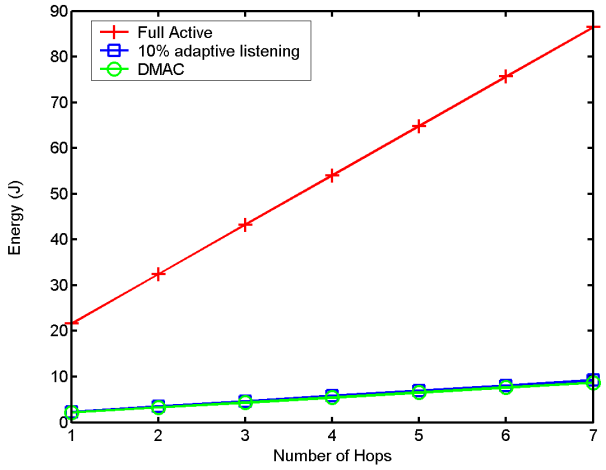



Figure 3.20: Total Energy consumption on each hop under low traffic load in Mote experiments.

long including physical layer header. In this light traffic, there is no queueing delay but only sleep latency. Figure 3.19 shows the mean packet latency with different hop length. Similar to the simulation results, the latency of both DMAC and full active CSMA/CA increase linearly with the number of hops with almost the same slop. The latency of DMAC is about 2100ms longer than the full active CSMA/CA which is about half of the total interval length. This is due to packet generated during the sleep period which have to be buffered until the active period. The 10% with adaptive listening scheme has a lower latency for the first two hops because of its shorter interval time. When the hop length is larger than 3, it has higher latency. Specifically, the latency has a jump each 2 hops. This is because by adaptive listening, a packet can be forwarded two hops instead of one hop in the 200ms active period. Each jump increases the latency about 2s which is exactly a schedule interval.

Figure 3.20 shows the energy cost with different hop lengths. Similar to the simulation results, the energy cost of all MAC increase linearly with the number of hops. However, the energy cost of the full active CSMA/CA increases much faster than other two MAC protocols. DMAC consumes slightly less energy cost than SMAC (10% with adaptive listening).

We also test the rate adaption of these MAC protocols. We vary the traffic load by changing the sensor report interval on the source node from 12s to 2s. The hop length is fixed at 5 hops. Figure 21 shows the averaged packet latency under different source report interval. Clearly full active CSMA/CA has the lowest latency. DMAC has a higher latency due to the initial latency at the source. SMAC(10% with adaptive listening), however, has even higher latency, especially when traffic load is heavy at small source report interval. The reason is that those packets suffered from both sleep delay and queuing delay. When traffic load is very high, contention would significantly increase packet latency as a retransmission

can only be done after one total schedule interval. The total synchronized active period of SMAC (10adaptive listening) has a smaller end-to-end capacity than DMAC.

## 3.5 Discussion

This chapter has proposed DMAC, an energy efficient and low latency MAC protocol for tree-based data gathering in wireless sensor networks. The major traffic in wireless sensor netowrks are from sensor nodes to a sink which construct a data gathering tree. DMAC utilizes this data gathering tree structure specific to sensor network applications to achieve both energy efficiency and low packet delivery latency. DMAC staggers the active/sleep schedule of the nodes in the data gathering tree according to its depth in the tree to allow continuous packet forwarding flow in which all nodes on the multihop path can be notified of the data delivery in progress and duty cycle adjustment command.

Data prediction is employed to solve the problem when each single source has low traffic rate but the aggregated rate at an intermediate node is larger than the basic duty cycle can handle. The interference between nodes with different parents could cause one traffic flow be interrupted because the nodes on the multihop path is not notified of the data transmission requirement. The use of an MTS packet is proposed to command nodes on the multihop path to remain active when a node fails to send a packet to its parent due to interference.

Our simulation and experimental results have shown that DMAC achieves both energy savings and low latency when used with data gathering trees in wireless sensor networks.

# Chapter 4

# DESS: Delay Efficient Sleep Scheduling

## 4.1 Overview

In the previous chapter, we investigated an approach to delay-efficient sleep scheduling, designed specifically for wireless sensor networks where the communication pattern is restricted to an established unidirectional data gathering tree. We showed that the sleep latency can be essentially eliminated by having a periodic receive-transmit-sleep cycle with level-by-level offset schedules. In this chapter we seek to address a more general and harder version of this problem: *how should the activity of sensor radio nodes be scheduled in arbitrary network communication topologies, in order to minimize the sleep latency while providing energy efficiency through periodic sleep?* This is clearly an issue of fundamental significance in the area of wireless sensor networks, and to our knowledge has never been investigated before. Unlike prior work in this area, which has focused primarily on designing new sensor network MAC protocols in an intuitive manner, we shall take an algorithmic approach.

The rest of the chapter is organized as follows: We first discuss the problem scenario and the assumptions made in this study (in section 4.2). We define a graph-theoretic combinatorial optimization problem formulation for delay efficient sleep scheduling (in section 4.3) for the single wake up schedule case where each sensor chooses exactly one of the $k$ slots to wake up. We show (in section 4.4) that this problem is in fact NP-hard in general. However, we are able to derive and analyze optimal solutions for some special cases, namely a ring topology and any tree topology. For arbitrary topologies, we propose several heuristics in section 4.5 and evaluate them using simulations in section 4.6.

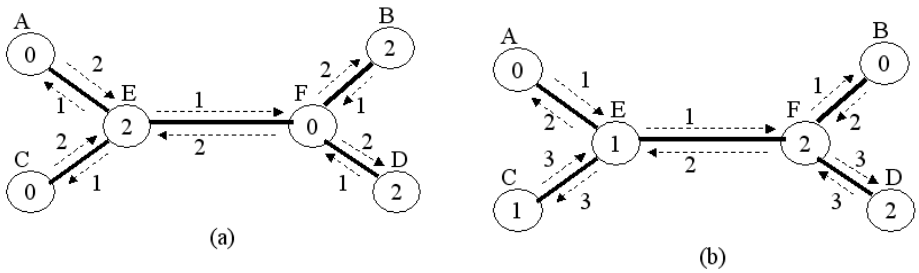



Figure 4.1: Examples of slot assignment with $k = 3$. The dotted arrows show the delay on each link in the corresponding direction.

## 4.2 Problem Scenario and Assumptions

In sensor networks with light traffic load, duty cycling (where sensors turn off their radios when not needed) is a very useful technique for reducing the energy consumption due to idle listening. We use $k$ as a parameter that captures the duty cycling requirements of an application. To achieve the requisite duty cycling, a sensor should be kept awake on an average for $\frac{1}{k}$ fraction of the time slots. We initially focus on the single wake up schedule case, where the schedule length is $k$ slots and each sensor is assigned one of the $k$ slots during which it activates its radio for reception (known as the active slot), while it can potentially transmit at any slot if it has a packet to be forwarded. If a node has to forward a packet to its neighbor, it can wake up at the active reception slot of that neighbor and transmit the packet. This conserves energy of both the transmitting and the receiving node. Figure 4.1 shows a couple of slot assignments on a network and the resulting delays on each link. Consider figure 4.1 (b). Assume that node A has a packet to send to node F. A would have received this packet in slot 0, but can only transmit to E at slot 1. Thus the delay from $A$ to $E$ is 1 (as $A$ waits for the complete reception of the packet at slot 0). Similarly E can only forward the packet to F in slot 2, thus incurring a delay of 1 from $E$ to $F$. In this case the end to end delivery latency is 2. Ideally, if every pair of nodes can have a path on which all nodes have sequentially increasing slots (modulo $k$), the latency will only be the number of hops between them times a single slot length ($\frac{1}{k}$-th of the schedule length). However a scheme such as the basic S-MAC scheme which synchronizes all nodes to have the same cycle will have a latency as large as the number of hops times the duration of a full period. As mentioned in section 4.1, DMAC can achieve the ideal case for any source to sink communication path for a unidirectional data gathering tree. However, this study addresses the issue of assigning slots to minimize the maximum delay between nodes that can communicate in an arbitrary pattern. Clearly as seen in figure 4.1, different slot assignments to the nodes in the network could result in significantly different path delays.

Before formally defining the problem, we describe our assumptions:

- **Synchronization:** None of the discussion about sleep scheduling would be relevant if there were not some mechanism to provide time synchronization in the sensor network. However, techniques capable of providing micro-second level synchronization have been developed for sensor networks [27, 39, 40].

- **Low Traffic:** We have assumed that there is very low traffic within the sensor network. This is reasonable in low-data-rate sensor networks where phenomena of interest occur rarely. Energy-efficient low-duty cycles are only possible if this assumption is true. It also justifies the fact that this problem formulation does not take into account any queueing latency due to congestion, or significant interference/collisions (though random access schemes may be implemented to handle occasional contention during the active periods, as in S-MAC). Since interference is not a primary concern in light traffic, we have not incorporated any local vertex/edge coloring constraints into our problem formulation which would be necessary for graph-coloring based TDMA scheduled access mechanisms such as [6].

- **Packet-Length Slot:** A related assumption we make is that for such a low traffic scenario each reception slot is of a fixed length that is sufficient only for the transfer of a single packet. Thus a packet may travel at most one hop in a single slot. Longer fixed slot lengths would not be energy-efficient if traffic is low.

- **Graph Abstraction:** While several recent papers in sensor networks (e.g. [41, 43, 42]) have shown that wireless links can be quite unreliable and vary significantly in packet reception rates in each direction, we have used a binary-link-based graph-theoretic problem formulation in this work. This is justified because the communication graph we are referring to is not necessarily the full wireless network, but a logical topology which can be constructed, for instance, by filtering or blacklisting out all unreliable/unidirectional links. Others have suggested that such blacklisting is necessary for reliable packet delivery in any case [43].

- **Arbitrary Communication Pattern:** In sensor networks where the traffic is restricted to data gathering from all nodes to a single sink, it is not necessary to minimize the delay diameter between any two nodes in the graph. However, this unidirectional traffic pattern is a special case which has been addressed previously in the DMAC work [46]. In more sophisticated embedded wireless sensor networks, which may involve complex patterns of in-network processing, or

communication between sensors as well as actuators, other traffic patterns are possible. We formulate the problem for the more general case in section 4.3.1, which as we shall show is in fact computationally harder. Although we do not treat it in depth here, an alternative formulation that can provide some way of weighing different application-specific traffic patterns is also defined in section 4.3.2.

- **Fixed Number of Slots:** In our formulation, we assume that the number of slots available to the network is fixed. This essentially defines the duration of the periodic sleep cycle, and the duty-cycle, which are assumed to be determined *a priori* by application-specific needs for energy efficiency as well as limitations on sleep/wakeup times of the radio hardware involved. As we shall see, generally with a larger number of available slots, the energy efficiency is higher but the end-to-end delay is also longer.

- **Energy Conservation:** Sensor node radios incur differing energy costs in idle listening, receiving and transmission modes. Transmission costs are generally higher than idle/reception costs. Technically, the minimum delay path obtained may involve longer hops (more transmissions) than the minimum hop-count path on the original graph. Thus delay minimization can result in a slight increase in the energy costs, however we believe this is a second order effect since the bulk of the energy savings in the network are provided by the sleep mode of the radio.

In section, 4.3, we formally define the problem of assigning slots to nodes to minimize the network delay.

## 4.3 Problem Definition

Let $G = (V, E)$ be an arbitrary graph. Let $k$ be the parameter that dictates the duty cycling requirements. As mentioned in section 4.2, we initially focus on the single wake up schedule case where the schedule length is $k$ slots and each sensor is assigned one of these $k$ slots. Assigning a slot $s \in [0 \cdots k-1]$ to a node $i$ schedules $i$ to wake up (activate its radio for receiving) only at slot $s$. While $i$ can transmit at any slot, it can only receive data at the beginning of slot $s$. Let $f : V \rightarrow [0 \cdots k-1]$ be a slot assignment function that assigns a slot to every node in the graph. Clearly $f$ determines the delay incurred in

transmitting data from one node to the other. For a given $f$, let $d_f(i,j)$ be the delay in transmitting data from $i$ to $j$ where $(i,j) \in E$:

$$d_f(i,j) \;=\; \begin{cases} k \;\; (\text{if } f(i) = f(j)) \\ (f(j) - f(i)) \;\bmod\; k \;\; (\text{otherwise}) \end{cases} \tag{4.1}$$

From the definition above, it also follows that:

$$d_f(i,j) + d_f(j,i) \;=\; \begin{cases} k \;\; (\text{if } f(i) \neq f(j)) \\ 2k \;\; (\text{otherwise}) \end{cases} \tag{4.2}$$

Delay on a path $P$ under a slot assignment $f$ is defined as

$$d_f(P) \;\;=\;\; \sum_{(i,j)\in P} d_f(i,j) \tag{4.3}$$

As seen from the above discussion, duty cycling requirements will lead to increased delays in the network. We consider the following scenarios:

### 4.3.1 All to All Communication

In this scenario, every pair of sensors is equally likely to communicate. Hence, it is desirable to assign slots to the nodes such that no two nodes incur arbitrarily long delays in communication. We characterize this network wide delay using the following definition:

***Definition 1:*** **Delay diameter** $(D_f)$: For a given graph $G = (V,E)$, number of slots $k$ and a slot assignment function $f : V \rightarrow [0 \cdots k-1]$, the *delay diameter* is defined as $\max_{i,j\in V} P_f(i,j)$, where $P_f(i,j)$ is the delay along the shortest delay path between nodes $i$ and $j$ under the given slot assignment function $f$.

In figure 4.1(a), the *delay diameter* is $5$, while in (b) it is $8$ (path D-F-E-C). Thus, in *all to all* communication, our design goal is given as follows:

***Definition 2:*** **Delay Efficient Sleep Scheduling (DESS):** Given a graph $G = (V,E)$ and the number of slots $k$, find an assignment function $f : V \rightarrow [0 \cdots k-1]$ that minimizes the *delay diameter* i.e.

$$f \;\;=\;\; \arg\min_{f'} \{D_{f'}\} \tag{4.4}$$

### 4.3.2 Weighted Communication

In this scenario, the frequency of communication between a pair of sensors is not the same across all pairs. This may happen in the case of a hierarchical network structure (like clustering). Here, it would be of interest to minimize the average delay in the network, which is defined as follows:

***Definition 3:*** **Average Delay diameter** ($D_f^{avg}$): For a given graph $G = (V, E)$, number of slots $k$, a slot assignment function $f : V \rightarrow [0 \cdots k-1]$ and weights $w(i,j) \geq 0$, the *average delay diameter* is defined as $\sum_{i,j \in V} w_{ij} * P_f(i,j)$, where $P_f(i,j)$ is the delay along the shortest delay path between nodes $i$ and $j$ under the given slot assignment function $f$.

In *weighted communication*, our design goal is the following:

***Definition 4:*** **Average Delay Efficient Sleep Scheduling (ADESS)** Given a graph $G = (V, E)$, the number of slots $k$, weights $w(i,j) \geq 0$, find an assignment function $f : V \rightarrow [0, \cdots k-1]$ that minimizes the *average delay diameter* i.e.

$$f \quad = \quad \arg\min_{f'} \{D_{f'}^{avg}\} \tag{4.5}$$

Intuitively, in both DESS and ADESS, the objective is to color a graph with the given $k$ colors such that the desired global objective (minimizing the *delay diameter* in the former and the *average delay diameter* in the latter) is achieved. The reader may perceive a connection to the well-known NP-complete graph coloring problem [37], which deals with minimizing the number of colors needed to ensure that no two adjacent vertices are colored the same. However, a key difference between the graph coloring problem and DESS (or ADESS) is that the former is essentially about a local constraint (adjacent vertices requiring distinct colors), while the latter is inherently more global in nature: adjacent vertices may share the same slot assignment but the maximum of the shortest delay paths between *all pairs* of nodes must be reduced. We will show below that both DESS and ADESS are also NP-Complete.

## 4.4 Analysis

DESS and ADESS are shown to be NP-hard in [49] by Narayanan Sadagopan. In this thesis, we formally characterize the optimal solution for DESS in two specific topologies (tree and ring),. We then show how the optimal solution for a ring may form a basic building block for an optimal assignment for cyclic graphs using the grid topology as an example.

### 4.4.1 Optimal Assignment on Specific Topologies

In this section, we formally characterize the optimal assignment function $f$ (that minimizes the *delay diameter* $D_f$ ) for 2 specific topologies: tree and ring. Using results from simulated annealing on a grid, we also show how an optimal assignment for a ring might form a basic building block of a good assignment on cyclic graphs.

#### 4.4.1.1 Optimal Assignment on a Tree

***Theorem** 1:* Consider a tree $T = (V, E)$. Let the number of slots be $k$. Let the diameter of $T$ (in hops) be $h$ (from node $a$ to $b$, say). Then for every slot assignment $f : V \rightarrow [0, \cdots k-1]$, $D_f \geq \frac{hk}{2}$.

*Proof:* Consider a path between two nodes $p$ to $q$ having $x$ hops. Since $T$ is a tree, this is the only path between $p$ and $q$. Consider an arbitrary slot assignment function $f : V \rightarrow [0, \cdots k-1]$. Now,

$$\begin{aligned} d_f(p \rightarrow q) &= \sum_{j=1}^{x} d_f(i_j, i_{j+1}) \\ d_f(q \rightarrow p) &= \sum_{j=1}^{x} (k - d_f(i_j, i_{j+1})) \end{aligned}$$

Thus,

$$\begin{aligned} d_f(p \rightarrow q) + d_f(q \rightarrow p) &= kx. \\ \max\{d_f(p \rightarrow q), d_f(q \rightarrow p)\} &\geq \frac{kx}{2} \end{aligned} \tag{4.6}$$

This is true for each pair of nodes including $a$ and $b$. Thus, for every slot assignment function $f$, $D_f \geq \frac{hk}{2}$, where $h$ is the diameter of $T$. ■

Based on theorem 1, the following assignment function $f$ will minimize the *delay diameter* of the tree $T = (V, E)$ whose hop diameter is $h$ (from $a$ to $b$): Just use 2 slot values, 0 and $\lceil \frac{k}{2} \rceil$. Let $d_f(a) = 0$. Adjacent vertices are assigned different slots (similar to a chess board pattern). In this case $\forall i, j : (i, j) \in E : \max\{d_f(i, j) = d_f(j, i)\} = \lceil \frac{k}{2} \rceil$. Hence $\max\{d_f(a \rightarrow b), d_f(b \rightarrow a)\} = \lceil \frac{hk}{2} \rceil$, which tightly matches the lower bound on the *delay diameter* of $T$. Thus, an optimal slot assignment for a tree balances the delay in each direction along a path as shown in figure 4.1(a).

#### 4.4.1.2 Optimal Assignment on a Ring

We first show the optimal assignment for the case where the number of nodes $n$ on a ring is a multiple of the number of slots $k$ i.e. $n = mk$. We then present a lower bound for the case when the number of nodes is not an exact multiple.

***Theorem** 2:* Consider $n = mk$ nodes $0, 1, \cdots mk-1$ arranged on a ring in the clockwise direction. The optimal slot assignment function $f$ is specified as follows: $f(0) = 0$. $\forall i : 1 \leq i \leq mk-1 : f(i) = (f(i-1)+1) \mod k$.

*Proof:* We will refer to such an $f$ as the sequential slot assignment as it assigns a sequentially increasing slot (modulo k) to the nodes around the ring (see figure 4.7 (a)). We prove theorem 2 by contradiction. For $k = 2$, it is easy to show that assigning 2 adjacent nodes the same slot incurs a delay of 2 in both directions on that link, while a sequential assignment will yield a delay of 1 in either direction. Hence, we focus on the case where $k \geq 3$. For a sequential slot assignment $f$, it is easy to show that the *delay diameter* is given by:

$$D_f = m(k-1) \tag{4.7}$$

Assume that there exists a slot assignment function $f'$, such that $D_{f'} < D_f$. In the rest of the proof, we will focus on the delay in the ring due to $f'$.

Consider a block of $m$ links on the ring from node 0 to node $m$ as shown in figure 4.2. Since we assumed that $D_{f'} < m(k-1)$, the shortest delay path from node 0 to node $m$ (and vice versa) must lie completely within the block. The alternative path has $m(k-1)$ links each incurring a delay of at least 1 (If this alternative path is the shortest delay path, it contradicts our assumption that $D_{f'} < m(k-1)$). This is true for every block of $m$ links on the ring. Figure 4.3 shows the shortest delay path for nodes within each of $k$ such blocks.

$\forall i : i \in [1,k], \forall j : j \in [1,2]$, let $d_{i1}$ be the delay in block $i$ from node $(i-1)m$ to $im$, while $d_{i2}$ be the delay in block $i$ from node $im$ to $(i-1)m$ as shown in the figure 4.3. We claim that $d_{min} = \min_{i,j}\{d_{ij}\} < 2m$. This can again be proved by contradiction as follows:

Consider a path from node 0 to node $\frac{k-1}{2}m$. There are two possibilities as shown in figure 4.3:

1. $0 \rightarrow m \rightarrow 2m \cdots \rightarrow \frac{k-1}{2}m$. The delay along this path is at least $\frac{k-1}{2}d_{min}$.
2. $0 \rightarrow mk-m \cdots \rightarrow \frac{k-1}{2}m$. The delay along this path is at least $\frac{k+1}{2}d_{min}$

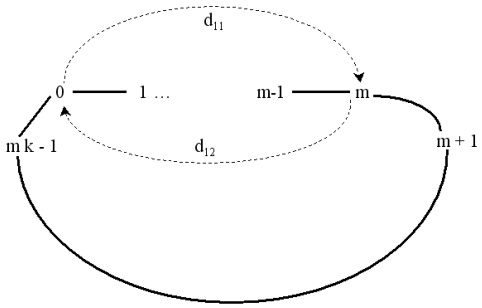



Figure 4.2: Shortest delay path for a single block of $m$ links.

Thus, if $d_{min} \geq 2m$, it contradicts the assumption that $D_{f'} < m(k-1)$. Moreover, since each block has $m$ links, each incurring a delay of at least 1,

$$m \leq d_{min} < 2m$$

Let $d_{min} = m + x$, where $x \in [0, m)$. Consider the block that has the lowest delay $d_{min}$. Without loss of generality, label the starting and ending node in this block as $mk - m$ and 0 as shown in the figure 4.4. Consider a path from node 0 to node $mk - m - x$. There are two possibilities as shown in figure 4.4:

1. $0 \rightarrow mk-m \rightarrow \cdots \rightarrow mk-m-x$. Delay along this path is at least $mk-d_{min}+x = m(k-1)$, which contradicts our assumption about $D_{f'} < m(k-1)$.

2. $0 \rightarrow m \rightarrow 2m \cdots \rightarrow mk - m - x$. Delay along this path is given by:

$$\begin{aligned} D &\geq \sum_{i=1}^{k-2} d_i + (m - x) \\ &\geq (k-2)(m+x) + m - x \\ &\geq m(k-1) + x(k-3) \\ &\geq m(k-1)(\text{for } k \geq 3) \end{aligned} \tag{4.8}$$

   This again contradicts our assumption that $D_{f'} < m(k-1)$.

Thus, for the ring with $n = mk$ nodes, the sequential assignment minimizes the *delay diameter*. ■

For the case when $n = mk + t$, for $0 < t < k$, the optimal solution is slightly more involved.

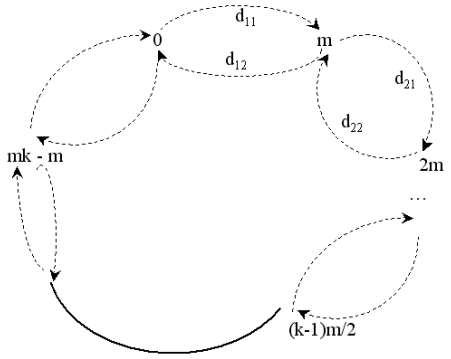



Figure 4.3: Shortest delay path for $k$ blocks of $m$ links each.

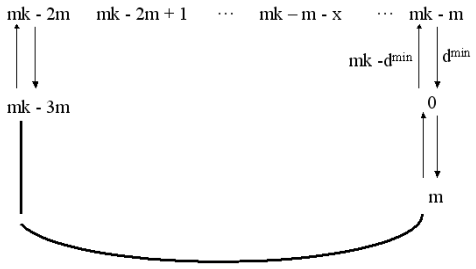



Figure 4.4: Paths from node 0 to node $mk - m - x$

***Theorem** 3:* For a ring with $n$ nodes where $n = mk + t$, for $0 < t < k$, the following is a lower bound on the delay diameter:

$$D_f \quad \geq \quad (m+1)k - \lfloor \frac{(m+1)k - y}{x} \rfloor \tag{4.9}$$

where $n = mk + t = (m+1)x + y$.

First we prove two lemmas that will be used in the theorem proof.

***Lemma** 1:* Consider $n = mk+t, 0 < t < k$. $C_1 = \sum_i d_f(i, i+1) + d_f(n-1, 0) = M \cdot k$, where $i \in [0, n-1]$, $M \geq m+1$.

*Proof:*

Since, there are $mk + t$ links and the delay on each link under any slot assignment is at least 1, $M$ has to be at least $m + 1$. Now,

$$d_f(i,j) = (S_j - S_i) \bmod k = \begin{cases} S_j - S_i & \text{if } S_j - S_i > 0 \\ S_j - S_i + k & \text{if } S_j - S_i \leq 0 \end{cases}$$

So:

$$\begin{aligned} C_1 &= \sum_i d_f(i, i+1) + d_f(n-1, 0) \\ &= (S_1 - S_0) \bmod k + ... + (S_0 - S_{n-1}) \bmod k \\ &= M \cdot k + (S_1 - S_0 + S_2 - S_1 + ... + S_0 - S_{n-1}) \\ &= M \cdot k \end{aligned}$$

It is easy to know that $C_2 = \sum_i d_f(i, i-1) + d_f(0, n-1) = M \cdot k$, where $i \in [1, n]$, $M \geq m+1$. $C_1 + C_2 = (mk+t)k$. Without loss of generality, let $C_1 \leq C_2$, then $M \leq \frac{mk+t}{2}$.

■

***Lemma** 2:* For an optimal slot assignment, $M = m + 1$.

*Proof:* It can be shown that the sequential slot assignment which assigns a sequentially increasing slot (by one and modulo k) has a *delay diameter* of $(m+1)(k-1)$. We will show that for $M > m+1$, the *delay diameter* will always be larger than $(m+1)(k-1)$. Assume $M = m+2$, hence $C_1 = (m+2)k$. We break the ring into blocks of size $m + 2$:

$$\begin{gathered} 0 \to 1 \to 2 \ldots m+1 \to m+2 \\ 1 \to 2 \to 3 \ldots m+2 \to m+3 \\ \vdots \\ mk+t-1 \to 0 \to 1 \ldots m \to m+1 \end{gathered}$$

Let $d_i$ be the sum of the all the link delays of the block starting at node i.

$$\begin{aligned} \sum_{i=0}^{mk+t-1} d_i &= (m+2) \cdot C_1 \\ &= (m+2)(m+2)k \end{aligned}$$

Let $d_{min}$ be the minimum of all $d_i$s, thus:

$$\begin{aligned} (mk+t)d_{min} &\leq (m+2)(m+2)k \\ d_{min} &\leq \frac{(m+2)(m+2)k}{mk+t} \\ d_{min} &\leq m+2+\frac{(2k-t)(m+2)}{mk+t} \end{aligned}$$

Consider the block that has the lowest delay $d_{min}$. Without loss of generality, let $d_0 = d_{min} = m + 2 + x$ shown in figure 4.5, where $x = \lfloor \frac{(2k-t)(m+2)}{mk+t} \rfloor$. Consider the path from node $m + 2$ to node 0. There are two possibilities:

1. $m + 2 \rightarrow m + 3 \cdots \rightarrow mk + t - 1 \rightarrow 0$. The delay along this path is $(m + 2)k - d_0$.

2. $m + 2 \rightarrow m + 1 \cdots \rightarrow 1 \rightarrow 0$. The delay along this path is also $(m + 2)k - d_0$.

For both case, the delay D is given by:

$$\begin{aligned} D &= (m+2)k - d_0 \\ &= (m+2)k - (m+2+x) \\ &= (m+1)(k-1) + k - 1 - x \end{aligned}$$

Since $M \leq \frac{mk+t}{2}$, when $M = m + 2$:

$$\frac{m+2}{mk+t} \leq \frac{1}{2}$$

Also because $0 < t < k$ and $k \geq 3$:

$$\lfloor \frac{2k-t}{k-1} \rfloor \leq 2$$

So:

$$\begin{aligned} \lfloor \frac{m+2}{mk+t} \cdot \frac{2k-t}{k-1} \rfloor &\leq 1 \\ x = \lfloor \frac{2k-t}{mk+t}(m+2) \rfloor &\leq k-1 \\ k-1-x &\geq 0 \\ D &\geq (m+1)(k-1) \end{aligned}$$

Thus we have proved that when $C_1 = (m+2)k$, the *delay diameter* will be at least $(m+1)(k-1)$. Similarly, it can be proved that for any $M \geq (m + 2)$, the *delay diameter* will be no smaller than $(m + 1)(k - 1)$.

Hence for an optimal slot assignment, $C_1 = (m + 1)k$. ■

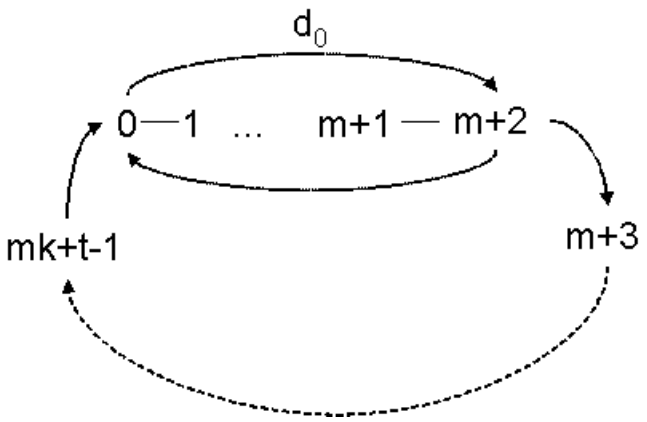



Figure 4.5: Paths from node $m + 2$ to node $0$

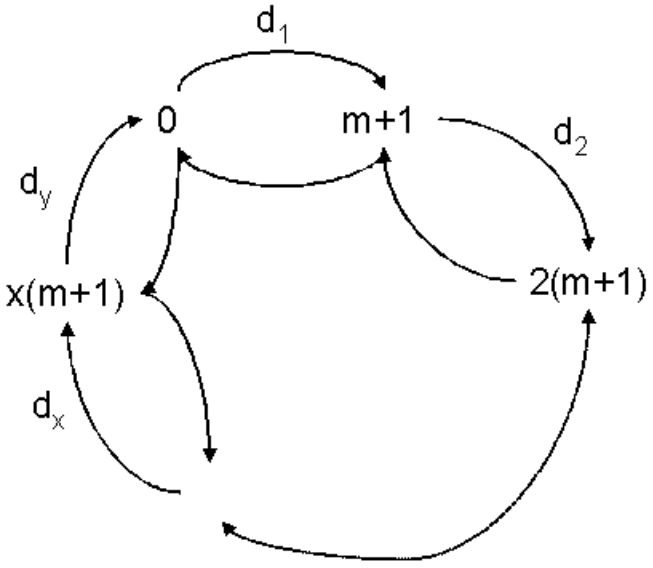



Figure 4.6: Shortest delay for $x$ blocks of $m + 1$ links each

Now we will calculate the lower bound on the *delay diameter* of the ring when $n = mk+t$. Similarly as the case when $n = mk$, we break the ring into blocks of size $m + 1$ shown in figure 4.6.

$$n \quad = \quad mk + t = (m + 1)x + y \tag{4.10}$$

where $0 \leq y < m + 1$

For any possible such block of $m + 1$ links, let $d_{min}$ be the minimum delay. The *delay diameter* of the ring is $(m + 1)k - d_{min}$. If we get the maximum value of $d_{min}$, we then achieve the smallest diameter $D = (m + 1)k - \max(d_{min})$.

Since $\sum d_i = (m + 1)k$, we have:

$$\begin{aligned} x \cdot d_{min} + d_y &\leq (m + 1)k \\ d_{min} &\leq \frac{(m + 1)k - d_y}{x} \leq \frac{(m + 1)k - y}{x} \\ \max(d_{min}) &= \lfloor \frac{(m + 1)k - y}{x} \rfloor \end{aligned}$$

Thus, the lower bound on the *delay diameter* $D_f$ for any slot assignment function $f$ is given by:

$$D_f \quad \geq \quad (m + 1)k - \lfloor \frac{(m + 1)k - y}{x} \rfloor$$

A slot assignment that achieves this lower bound is illustrated by the figure 4.7 (b).

In section 4.5, we describe some centralized and distributed heuristics for slot assignment on general topologies.

## 4.5 Heuristic Approaches

From the theoretical analysis, we know that DESS is NP-hard, hence it is unlikely that there exist polynomial time algorithms for solving it. We instead propose several heuristic solutions in this section and evaluate their performance through simulations in section 4.6.

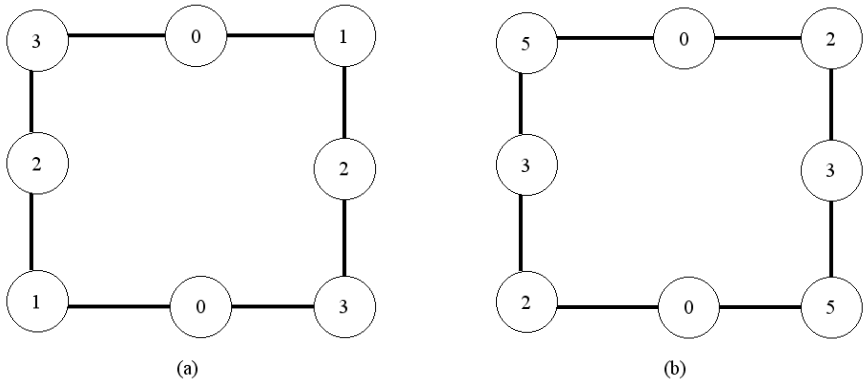



Figure 4.7: (a) The sequential slot assignment $f$ obtained for a ring with $n = 8$ nodes and $k = 4$ slots $(n = mk)$. Here $D_f = 6$. (b). A slot assignment $f$ obtained for a ring with $n = 8$ nodes with $k = 6$ using the optimal construction for $(n = mk + t)$. Here $D_f = 9$ which matches the lower bound in equation 4.9.

### 4.5.1 Centralized Algorithm

Initially, all nodes are assigned the same slot and the *delay diameter* $D$ of the network is computed. By either deterministic or random order, each sensor node calculates the *delay diameter* of the network for all possible slot assignments for itself while keeping other nodes' slots unchanged. If the minimum of the *delay diameter*s of all possible slot assignments $d_{min}$ is smaller than the previous *delay diameter* ($d_{min} < d$), the node changes its slot to the one that gives the minimum *delay diameter* and updates $d \leftarrow d_{min}$. If the *delay diameter* is unchanged, it chooses the new slot or keeps the current slot with equal probability. Otherwise it keeps its current slot unchanged. After all nodes finish this operation, the iteration can be repeated again. The number of iterations depends on limitations on the algorithm duration (which in turn depends upon the size of the network). The pseudo code for the centralized algorithm is shown below.

**Algorithm** *Centralized*

1. Assign slot 0 to all nodes in G
2. $d = D(G)$ $//delay\ diameter\ \ of\ \ G$
3. **for** $i \leftarrow 1$ **to** $n$ //number of iterations
4. **for** each node $s$ in the network
5. **for** $k_1 \leftarrow 0$ **to** $k - 1$ //total slots
6. $ok \leftarrow slot(s)$
7. $slot(s) \leftarrow k_1$
8. $md \leftarrow D(G)$

9. **if** $d_{min} < d$
10. **then** $d \leftarrow d_{min}$
11. $minslot \leftarrow k_1$
12. **if** $d_{min} == d$
13. **then** $minslot \leftarrow k_1$ with 50% probability
14. $minslot \leftarrow ok$ with 50% probability
15. $slot(s) \leftarrow minslot$

### 4.5.2 Localized Algorithms

The centralized algorithm assumes complete knowledge of the network topology and slot assignment. In this section we consider some localized algorithms in which a sensor node only knows the information stored at its neighbors.

We propose two different localized algorithms.

- Local-Neighbor: A node knows only the slot assignments of its neighbors. It chooses a slot which minimizes the maximum of its delays to and from its immediate neighbors. This can be repeated for a certain number of iterations.
- Local-DV: Its working is similar to Distance Vector routing techniques. Each node maintains two Distance Vector tables: a forward table $FDV$ which stores its shortest delays *to* all other nodes and a backward table $BDV$ which stores its shortest delays *from* all other nodes. These two tables can be calculated using the basic Bellman-Ford technique. A sensor node also knows the DV tables of its direct neighbors. A sensor node calculates the DV tables for all possible new slot assignments for itself. Let the maximum value of entries in the sets of the two DV tables over all possible slot assignments be $maxd$. The node will choose the slot which gives the minimum $maxd$.

The pseudo codes of Local-Neighbor and Local-DV are shown below.

**Algorithm** *Local-Neighbor*

1. Each node $s$ get the slots of its direct neighbor $N(s)$
2. $mind \leftarrow MAX\_VALUE$
3. **for** $k_1 \leftarrow 0$ **to** $k-1$ //total slots
4. $slot(s) \leftarrow k_1$
5. $fd(s,t) \leftarrow$delay from $s$ to $t$ in $N(s)$

6. $bd(s,t) \leftarrow$delay from $t$ in $N(s)$ to $s$

7. $maxd \leftarrow max(fd, bd)$

8. **if** $maxd < mind$

9. **then** $mind \leftarrow maxd$

10. $minslot \leftarrow k_1$

11. $slot(s) \leftarrow minslot$

**Algorithm** *Local-DV*

1. Each node $s$ calculate DV tables $FDV, BDV$

2. Get the $FDV, BDV$ of its direct neighbor $N(s)$

3. $mind \leftarrow MAX\_VALUE$

4. **for** $k_1 \leftarrow 0$ **to** $k-1$ //total slots

5. $slot(s) \leftarrow k$

6. update $FDV, BDV$

7. $maxd \leftarrow max(FDV, BDV)$

8. **if** $maxd < mind$

9. **then** $mind \leftarrow maxd$

10. $minslot \leftarrow k_1$

11. $slot(s) \leftarrow minslot$

### 4.5.3 Randomization

The simplest slot assignment is to just randomly choose a slot for each node once. In a dense network where a node has a large number of neighbors (where multiple paths are available for any pair of nodes), there is a high probability that this assignment may lead to a short delay path. We call this decentralized random slot assignment as Random-Average. The performance of this method is evaluated by the expected value of the *delay diameter*. The randomized slot assignment can also be done in a centralized manner. We refer to this centralized version as the Random-Minimum strategy. After a certain number of iterations of choosing random slots for all the nodes, this strategy chooses the assignment that gives the minimum *delay diameter* and then deploys the slot assignment in the network.

While all the above heuristics can be used for any topology, we next propose a specialized heuristic for the grid that exploits the structure of the topology.

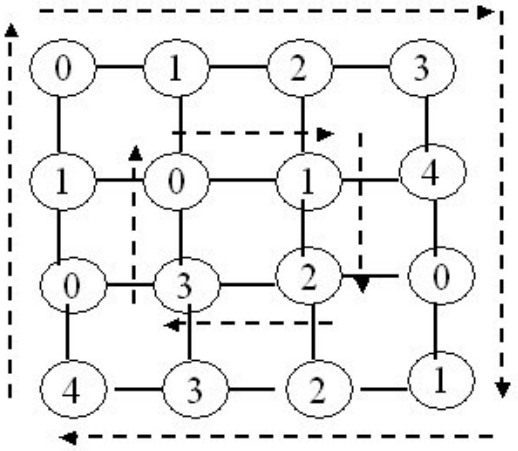



Figure 4.8: Concentric ring allocation for a grid of $4 \times 4$ nodes with $k = 5$. The dotted lines illustrate the concentric rings at each level.

### 4.5.4 Concentric Ring for the Grid topology

We believe that the optimal assignment on a ring can serve as a basis for a low latency assignment on a grid that can be viewed as a set of concentric rings with interconnecting bridges. The outer most ring is given a sequential assignment going in the clock-wise direction starting at 0. For every other ring, a slot assignment is chosen that offers the best *delay diameter* for that ring. An example of this assignment is shown in figure 4.8

## 4.6 Simulation Results

In this section, we evaluate the performance of the heuristic algorithms on the grid topology (in section 4.6.1) and random topology (in section 4.6.2) through high level simulations. Since the current study focuses on comparing the *delay diameter* only across these heuristics, even the distributed algorithms are simulated in a centralized manner (without analyzing their overhead). We also assume that the number of slots $k$ is dictated by the duty cycling requirements of the application.

### 4.6.1 Grid Network

First we evaluated the six schemes on a grid topology: Centralized, Local-DV, Local-Neighbor, Random-Avg, Random-Min and Concentric Ring. To have a fair comparison, the centralized and the two local schemes had the same number of iteration $I = 20$ while the random algorithms ran for $I \times k$ times.

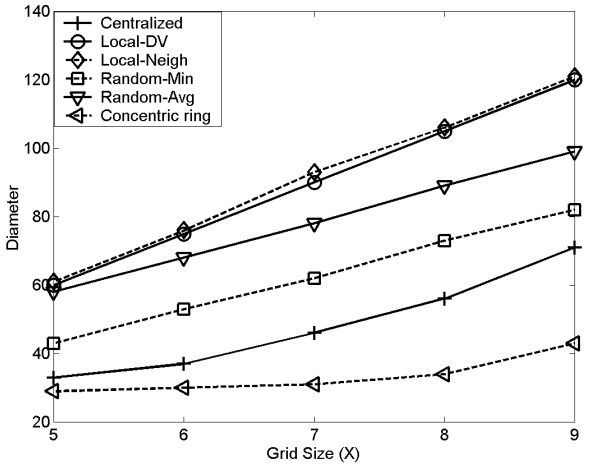



Figure 4.9: The *delay diameter* of the heuristic algorithms versus grid size for the number of slots fixed at $k = 15$. The grid is given as $X \times X$.

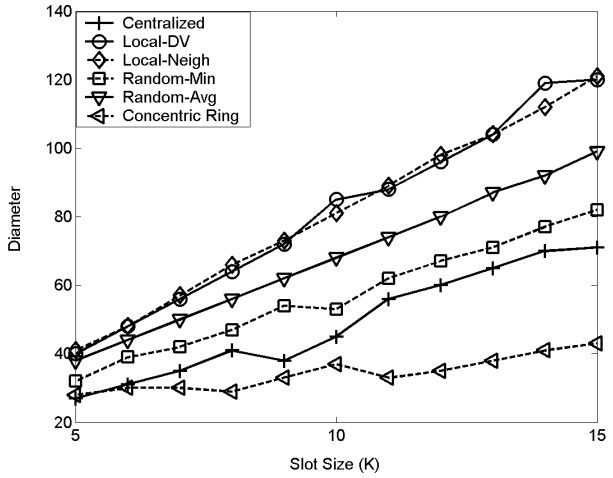



Figure 4.10: The *delay diameter* of the heuristic algorithms versus the number of slots ($k$) for a fixed grid size of $9 \times 9$.

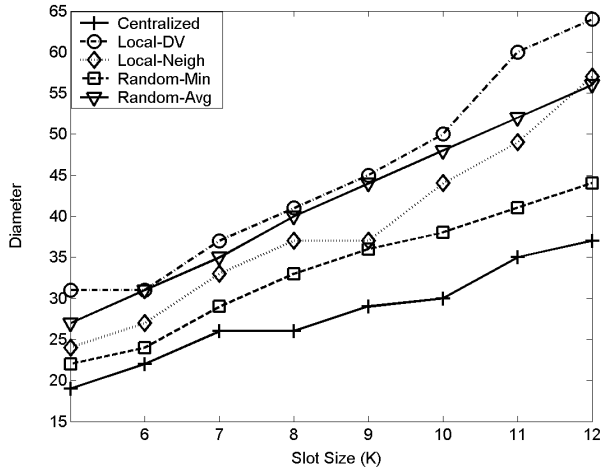



Figure 4.11: The *delay diameter* of the heuristic algorithms versus the number of slots ($k$) for nodes randomly deployed in a $10 \times 10$ area. The transmission range is 2.

Figure 4.9 shows the results for different grid sizes while the number of slots $k$ is fixed at 15. By exploiting the structure of the grid, concentric ring has the best performance compared to all other schemes. The centralized scheme is slightly worse than the concentric ring at small grid size but is about 2 times worse than concentric ring when grid size is large. Both the randomized schemes perform worse than the centralized algorithm with Random-Min doing better than Random-Avg. Moreover the two localized algorithms also seem to perform poorly compared to the centralized one. This is possibly because of the fact that the *delay diameter* of a network being a global property, the local optimization schemes do not converge to the global optimum. Overall, we find that the centralized scheme can reduce the *delay diameter* of random schemes by about 50%, while the concentric ring can provide a further reduction of about 50%.

Although $k$ should be decided by the duty cycle requirement of applications, it is interesting to see its impact on the *delay diameter*. Figure 4.10 shows the results for different values of $k$ while the grid size is fixed at $9 \times 9$. Clearly, the *delay diameter* increases almost linearly with the number of slots $k$. Concentric ring performs the best while local schemes perform the worst. We further evaluated these schemes on a larger grid with $20 \times 20$ nodes and values of $k$ up to 20. We observed similar trends in performance.

### 4.6.2 Random Network

We also tested the five schemes (excluding the Concentric Ring heuristic) on a network with randomly deployed sensor nodes.

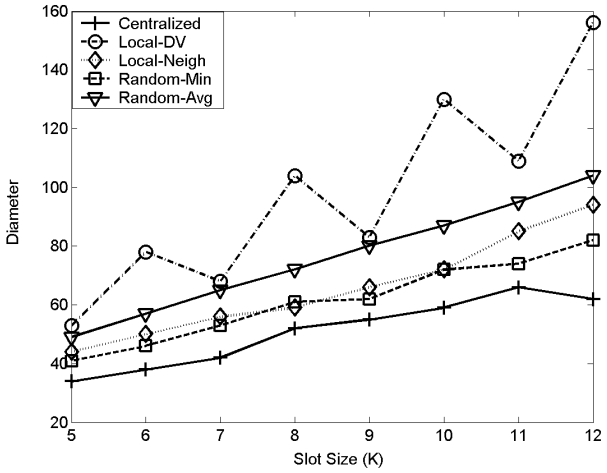



Figure 4.12: The *delay diameter* of the heuristic algorithms versus the number of slots ($k$) for nodes randomly deployed in a $3 \times 33$ area. The transmission range is 2.

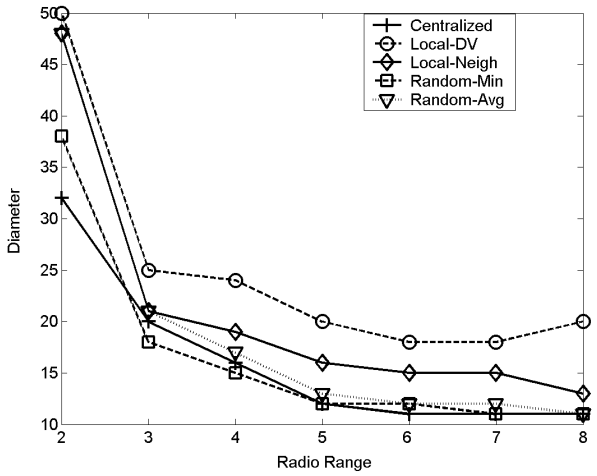



Figure 4.13: The *delay diameter* of the heuristic algorithms versus the radio transmission range for nodes randomly deployed in a $10 \times 10$ area. Number of slots is fixed at $k = 10$.

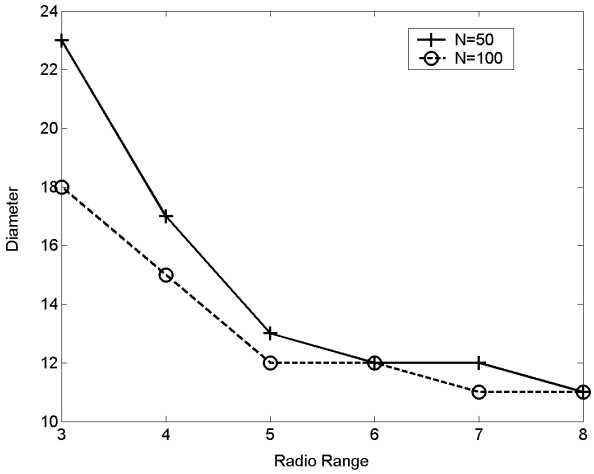



Figure 4.14: The *delay diameter* of the Random-Min algorithm versus radio transmission range for nodes randomly deployed in a $10 \times 10$ area with $N = 50$ and $N = 100$. The number of slots $k = 10$.

First we fixed the radio transmission range at 2. Figure 4.11 and 4.12 show the result with 100 nodes uniformly distributed in a $10 \times 10$ square and a $3 \times 33$ rectangle. In both cases, the centralized scheme performs best, followed by Random-Min. It is interesting to note that in the random network, Local-Neighbor now has a smaller *delay diameter* than Random-Avg. In the $3 \times 33$ area, the Local-Neighbor performs quite well even on comparison with the Random-Min scheme. We believe this is not because Local-Neighbor perform better but because Random schemes perform worse in a random graph. In a grid, each internal node has 4 direct neighbors. In a random graph, however there is a probability that some nodes are bottlenecks (nodes through which several paths go through). An improper slot assignment for such a bridge node may hurt the *delay diameter* significantly. In a $3 \times 33$ long rectangle, the probability of a node being a bridge becomes higher, which is likely the reason that the performance of Local-Neighbor is closer to the Random-Min scheme. This intuition is also backed by figure 4.14 which shows the *delay diameter* obtained by the random schemes with 50 and 100 nodes. With 100 nodes, the density and the average degree of the network increases, the random schemes have better performance because of the increased number of paths between any pair of nodes (and hence fewer bottlenecks).

Figure 4.13 shows the effect of the radio transmission range $R$ on the *delay diameter*. As $R$ increases, the *delay diameter* decreases. This is because an increase in $R$ decreases the graph diameter (in hops).

Thus, for the single schedule case, where each node chooses exactly one of the $k$ slots to wake up, we have presented several heuristics in section 4.5 and evaluated them through simulations in section 4.6.

## 4.7 Discussion

In this chapter we have addressed the important problem of minimizing communication latency while providing energy-efficient periodic sleep cycles for nodes in wireless sensor networks. The objective is to minimize the latency given the duty cycling requirement that each sensor has to be awake for $\frac{1}{k}$ fraction of time slots on an average. For the single wake up schedule case, where each sensor can wake up at exactly one of the $k$ slots, we have provided graph-theoretic problem formulations for arbitrary all-to-all (DESS) as well as weighted communication patterns (ADESS). We also proved that both these problems are NP-hard. We then focused on the DESS problem and derived and proved optimal solutions for two special cases, *viz.* the tree and ring topologies. For arbitrary topologies, we proposed several heuristics and evaluated them through simulations. These simulations reveal several interesting observations: that purely localized heuristics tend to perform worse than simple randomized slot allocations, that our centralized scheme can provide delay reductions of around 50% over randomized schemes and that specialized heuristics (that exploit the topological structure) like the concentric ring for the grid can provide additional gains.

Further, we showed that by carefully choosing multiple wake up slots, one can obtain significant savings in the latency at the same duty cycling. Using this technique, we propose algorithms with provable guarantees on tree, grid and arbitrary graphs. These results obtained from an algorithmic perspective are novel and quite different from prior work in this area which has focused primarily on intuitive MAC protocol designs (such as S-MAC [60], T-MAC [26], and our own work on D-MAC [46]).

# Chapter 5

# MLSR: Minimum Latency Joint Scheduling and Routing

## 5.1 Overview

In the last chapter (DESS), we investigated the problem of minimizing the worst case communication latency for arbitrary all possible source-destination pairs while each node has a fixed duty cycle. In many sensor network application scenarios, typical communication patterns are from sensors sources to several limited number of base stations. All sinks are same, which means a packet can be routed to any one of the sink. Also at any specific time, not all sensor nodes have packets to sent to the base stations. So it is not necessary to minimize the worst case latency for arbitrary all-to-all communication pairs. Instead, we need only optimize for the existing flows. An example is shown in figure 5.1

The best way to save energy is to put a node to sleep. However, a node that is off is unable to deliver the sensor data to the base stations. This creates a fundamental trade-off. As the flows in sensor network are likely to be predictable long-lived, we could schedule the on-off activity of the sensor nodes that can wake up to help deliver the sensor data reports in time and go to sleep to save energy otherwise.

Putting nodes to sleep affects another important communication layer: the network layer. A node in sleep is no longer part of the network. Therefore, by the sleep scheduling, the topology of the network keeps changing at different times. A link between two neighboring nodes is available only if both of them are scheduled to be active at the same time slot. The paths selected by the routing algorithm also affects latency and power consumption. Different links could have different latency depending on the scheduled sending and receiving slots of the nodes. A shortest hop path may not be the path with shortest latency.

Thus, given certain source nodes and base stations in a sensor network, there are two key design considerations: one is the scheduling, the other is routing. Those two are closely coupled together and will affect each other. For example, in figure 5.1, node J needs to allocate time slots for data reports from

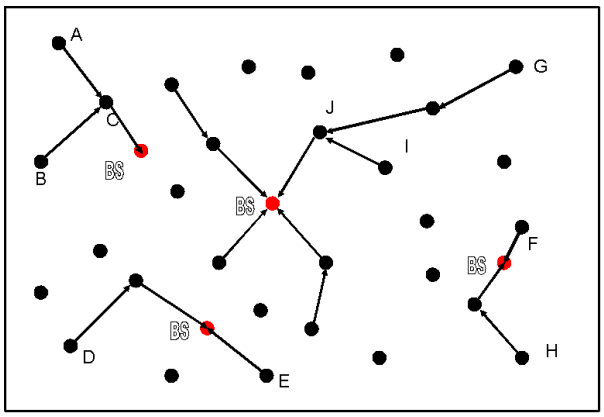



Figure 5.1: A data gathering application in a wireless sensor network.

G and I. However if node I changes its next hop to other node or it does not generate any more reports, then node J only needs to schedule slots for reports from G. There are three possible approaches:

1. Scheduling first: Determine the schedules of the sensor nodes first. Based on the schedules, using a routing algorithm to find an energy efficient and low latency path.

2. Routing first: Using a routing algorithm to find a path first. Given the path, find the schedules of the nodes on the path.

3. Joint scheduling and routing: find the schedule and routing solution jointly.

Both scheduling first and routing first scheme have their disadvantages. Scheduling first schemes may cause routing scheme hard to find short paths while routing first schemes may cause low latency schedule impossible. In this chapter, we will discuss the joint scheduling and routing approach. Particularly, we are interested in finding paths and schedules that achieve the minimum average latency. We assume that energy efficiency will be provided automatically by the schedule, as nodes only wake up when needed and asleep during other times. This is clearly an issue of fundamental significance in the area of wireless sensor networks, and to our knowledge has never been investigated before.

## 5.2 Scheduling and Routing in Wireless Sensor Network

### 5.2.1 Application Scenario

Broadly speaking, there are two kinds of applications in sensor networks: event driven and continuous monitoring. In event driver sensor network, most of the time the sensor nodes are off until certain interesting event happens. Then the nodes begin to send data to base station. In a continuous monitoring

sensor networks, sensor nodes sample and transmit data at regular intervals requested by the base station. In both kinds of applications, at any specific time, only certain nodes are source nodes that need to sample environment and report to base station. We need to find paths and schedules of the nodes on the paths, in which the average latency of all the active flows are minimized.

We first describe the basic application assumptions.

1. **Radio:** Consider a static wireless sensor network, all nodes are equipped with a single radio with omni-directional antennas. The transmission power and data rate are fixed.

2. **Synchronization:** None of the discussion about TDMA link scheduling would be relevant if there were not some mechanism to provide time synchronization in the sensor network. However, techniques capable of providing micro-second level synchronization have been developed for sensor networks [27, 39, 40].

3. **TDMA time slots:** Time is divided into equal sized slots that are long enough for one packet transmission and grouped into frames. Thus a packet may travel at most one hop in a single slot. Some works on TDMA focus on minimizing the length of the frame subject to the constraint that every node or link is assigned at least one slot. In this work, however, the latency is not affected by the frame length but the sleep latency caused by the scheduling.

4. **Sources:** Certain source nodes sample the environment periodically. The period can be different for different source nodes. Each node generates certain number of packets of fixed length which need to be transmitted in one TDMA frame.

5. **Sinks:** There are several sink nodes in the network. A sensor report can be delivered and only need to be delivered to any one sink.

6. **Graph Abstraction:** Similar to the assumption in DESS, we assume the wireless network can be abstracted into a graph.

7. **Energy Conservation:** Same as in DESS, we assume the major energy conservation comes from turning radio off.

8. **Interference:** We will consider two different interference models: FDMA-based multiple channels and single channel.

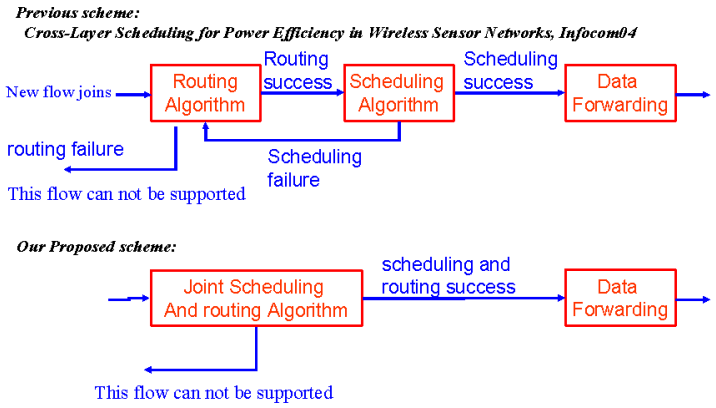



Figure 5.2: Two scheduling and routing schemes in wireless sensor networks.

### 5.2.2 Routing and Scheduling

In sensor networks with light traffic load, putting nodes to sleep (where sensors turn off their radios when not needed) is a very useful technique for reducing the energy consumption due to idle listening. We use $K$ as a parameter that indicates the number of slots in a TDMA frame. The $K$ is big enough that all periodically report will have at least one report every $K$ slots. In each slot, the radio is in one of the three states: transmitting, receiving or free. Radio should be put to sleep in free state to save energy. If a node has to forward a packet to its neighbor, it need to find a slot that both the neighbor and itself are free. Then the slot in the sender is marked as transmitting and marked as receiving in the receiver. This conserves the energy of both the transmitting and the receiving node. We do not consider slots reserved for routing setup and neighbor discovering.

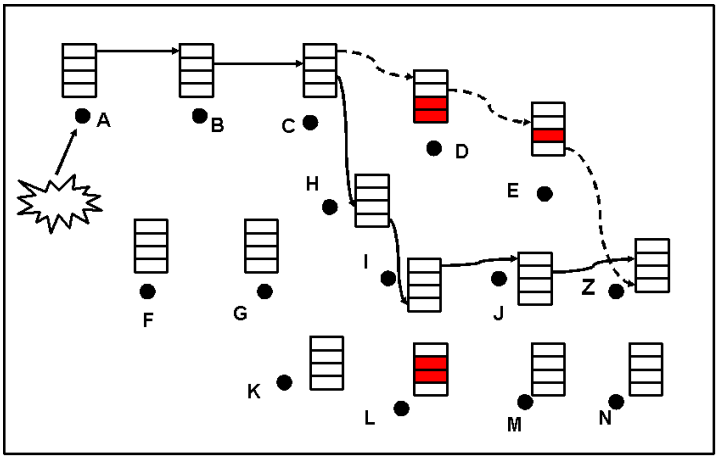
Let $G = (V, E)$ be an arbitrary graph. Let $P_m$ be the path for flow $m$ from a source to a sink. Let $f$ denotes the slot assignment. For a given node $i$, let $R_m^i$ and $S_m^i$ denote the receiving and transmitting slots of node $i$ for flow $m$ [1]. Clearly, $R_m^i$ and $S_m^i$ determine the delay incurred in transmitting data from one node to the other. Let $d_m(i, j)$ be the delay in transmitting data from $i$ to $j$ where $(i, j) \in E$:

$$d_m(i,j) = \begin{cases} S_m^i - R_m^j \ \ (\text{if } S_m^i - R_m^j > 0) \\ (S_m^i - R_m^j) + K (\text{otherwise}) \end{cases} \tag{5.1}$$

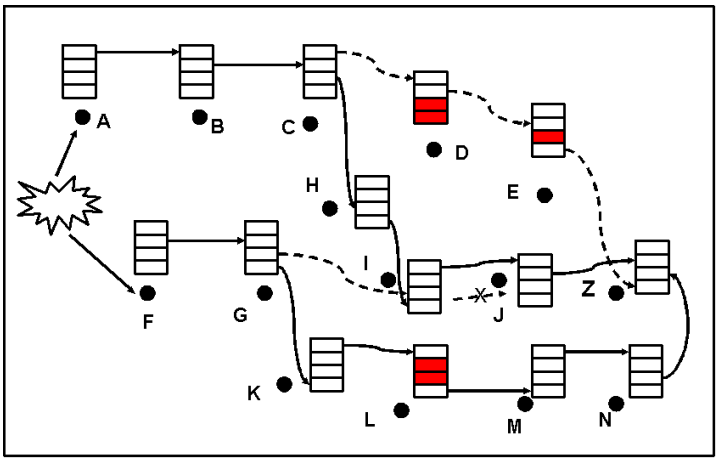
Delay on a path $P_m$ under the slot assignment is defined as

$$d(P_m) = \sum_{(i,j) \in P_m} d_m(i,j) \tag{5.2}$$

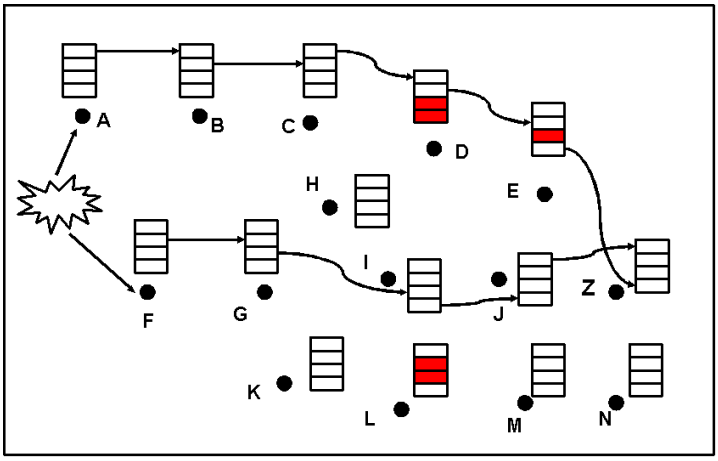
[1] $S_m^i = R_m^i$




(a) Scheduling and routing for 1st flow




(b) One possible scheduling and routing for two flows




(c) A better scheduling and routing for two flows

Figure 5.3: Example of scheduling and routing for two active flows.

As seen from the above discussion, the end-to-end delay for flow $m$ depends on both the path and the slot assignment.

**Active Flows Communication:** For a sensor network application, at any specific time, not all nodes have sampling data to report to the base station: there are only a limited number of active flows in the network. Here, it would be of interest to minimize the average delay of the active flows in the network, which is defined as follows:

***Definition** 5:* **Average Delay** ($D^{avg}_{P,f}$): For a given graph $G = (V, E)$, number of slots $k$, a slot assignment function $f$ and paths $P$ for $M$ flows, the *average delay* is defined as $\sum_{m \in M} d(P_m)$, where $d(P_m))$ is the delay along the path $P_m$ under the given slot assignment function $f$.

In *active flows communication*, our design goal is the following:

***Definition** 6:* **Minimum Latency joint Scheduling and Routing (MLSR)** Given a graph $G = (V, E)$, the number of slots $k$ and $M$ flows, find a path $P_m$ for each flow, and an slot assignment function $f$ that minimizes the *average delay* ($D^{avg}_{f}$) i.e.

$$f \quad = \quad \arg\min_{P',f'} \{D^{avg}_{P',f'}\} \tag{5.3}$$

Figure 5.2 shows an example of separate routing and scheduling solution. When there is a new flow request, the routing algorithm will find a route first, then on the given route, if a schedule is achievable, the following data forwarding will use the route and scheduling to forward data report. If the scheduling process fails, then the routing algorithm is asked to find a new route.

Figure 5.3 shows an example of scheduling and routing for two active flows. In figure 5.3(a), first there is only one active flow. If we run a shortest hop routing algorithm, it will find the path of $A \rightarrow B \rightarrow C \rightarrow D \rightarrow E \rightarrow Z$. A schedule can be assigned on the nodes on the path. The delay is 8. If using the link delay as the weight of the link, however, shortest path routing algorithm now select $A \rightarrow B \rightarrow C \rightarrow H \rightarrow I \rightarrow J \rightarrow Z$ which has a latency of only 6. Now assume there is another flow request from node F to node Z. Suppose the flow want to use route $F \rightarrow G \rightarrow I \rightarrow J \rightarrow Z$. However at link $(i, j)$, there is no slot that both $i$ and $j$ are free, so the flow is forced to choose route $F \rightarrow G \rightarrow K \rightarrow L \rightarrow M \rightarrow N \rightarrow Z$ which has a delay of 8. So the total delay for the two active flows is 14 in figure 5.3(b). However, as shown in figure 5.3(c), if source A use route $A \rightarrow B \rightarrow C \rightarrow D \rightarrow E \rightarrow Z$ and source F use route $F \rightarrow G \rightarrow I \rightarrow J \rightarrow Z$. A schedule can be achieved with total delay of only 12. This example shows that latency reduction by joint scheduling and routing.

Intuitively, in MLSR, the objective is to color a graph with the given $K$ colors such that the desired global objective (minimizing the *delay diameter* in the former and the *average delay diameter* in the

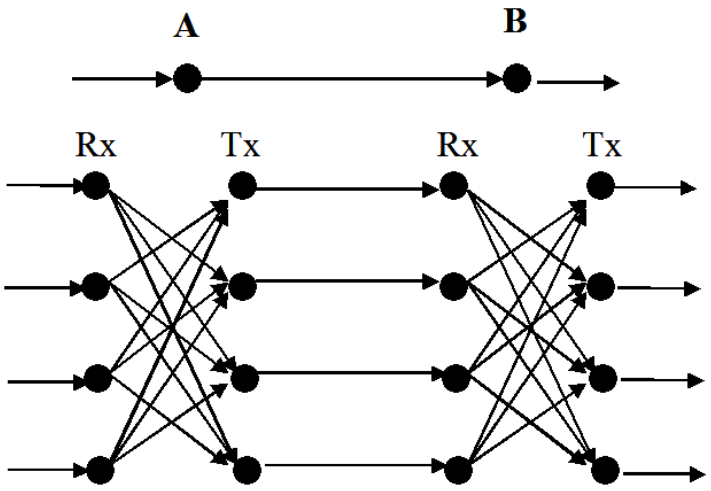



Figure 5.4: Delay Graph: Link

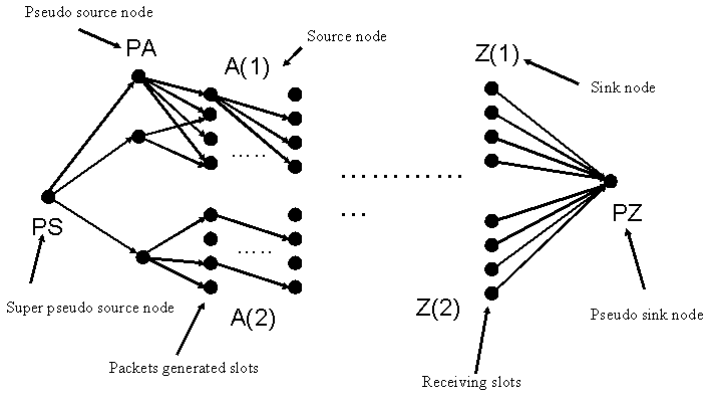



Figure 5.5: Delay Graph: Network

latter) is achieved. The reader may perceive a connection to the well-known NP-complete graph coloring problem [37], which deals with minimizing the number of colors needed to ensure that no two adjacent vertices are colored the same. However, a key difference between the graph coloring problem and MLSR is that the former is essentially about a local constraint (adjacent vertices requiring distinct colors), while the latter is inherently more global in nature: adjacent vertices may share the same slot assignment but the maximum of the shortest delay paths between *all pairs* of nodes must be reduced.

## 5.3 Minimum Latency Joint Scheduling and Routing

### 5.3.1 Delay Graph

Suppose there are $M$ source nodes and $N$ sinks. The number of slots per TDMA frame is $K$. We create a *Delay Graph* as following:

1. For each sensor node $u$, create $2K$ graph nodes. The first $K$ nodes represent the receiving slots of sensor node, $R_u^i$, which means node $u$ receiving a packet at slot $i$ from the previous hop. The second $K$ nodes represent the transmitting slots of the sensor node, $S_u^i$, which means node $u$ transmitting the packet at slot $i$ to the next hop.

2. From the each receiving node of $u$, $R_u^i$, add a link to all transmitting nodes of $u$, $S_u^j$ except $j = i$. The delay of the link is decided by equation 5.1;

3. If two sensor nodes $u$ and $v$ can communicate with each other, create a graph link from each transmitting node of $u$, $S_u^i$ to the corresponding receiving node of $v$, $R_u^i$, which means node $u$ transmits a packet to $v$ at slot $i$. Assign the weight of the link 1.

4. Add a super sink graph node $PZ$. Add a link from each receiving nodes $R_{Z(n)}^i$ of sink $Z(n)$ to $PZ$. Assign the weight of the link 1.

5. Add a graph node $PA(m)$ for each source sensor node $A(m)$, which is called pseudo source node. Add a graph link from $PA(m)$ to each receiving node $R_{A(m)}^i$ of $A(m)$, with link delay weight of 1. If a source have more than one packet to send per TDMA frame, treat each packet as a different source.

6. Add a super pseudo source graph node $PS$. Add a directed link from $PS$ to each $PA(m)$ with link delay weight of 1.

Figure 5.4 shows how to split a node to $2K$ nodes in the delay graph and the connection between two nodes. Figure 5.5 shows an example of a complete delay graph with two sources while source $A(1)$ have two packets per frame.

### 5.3.2 FDMA interference model

In this section, we consider the joint scheduling and routing problem under the FDMA interference model. Each link can use a FDMA channel for communication. We assume that the number of channels is large enough so that any two links within interference range are assigned two different FDMA channel. In this model, the constraints on the radio are that:

1. With a single omni-directional antenna, no node can transmit or receive data at the same time.
2. A node cannot transmit different packets to different receivers at the same time.
3. A node can only receive a packet from a single sender at one time.

We also assume no node will broadcast a single packet to multiple receivers. Therefore each sender can only have one receiver.

First we consider the problem of finding $M$ node-disjoint paths on the delay graph. We can map the $M$ node-disjoint paths on the delay graph to the minimum latency joint scheduling and routing solution for $M$ sources.

*Proposition 4:* The minimum weight $M$ node-disjoint paths in the delay graph can be mapped to a minimum latency joint scheduling and routing solution.

*Proof:* First we explain how to get a route and schedule from the node-disjoint disjoint path. Suppose one of the $M$ node-disjoint path is $P_m$. Assume the nodes the path goes through are: $PS, PA(m)$, $R^{i1}_{A(m)}, S^{j1}_{A(m)}, ..., R^{i_u}_u, S^{j_u}_u, R^{i_v}_v, S^{j_v}_v, ..., R^{i_{Z(n)}}_{Z(n)}, PZ$. It is easy to know that $j_u == i_v$. The route for the flow $m$ then is $A(m), ..., u, v, ...Z(n)$. The schedule of node $u$ on the route is to wake up to receive packet at $i_u$ and send it at $j_u$. At $j_u$, node $v$ is wake up to receive packet and $v$ will forward it to the next hop at time $j_v$.

Then we need to prove that the schedule we get satisfy the three constraints of the radio. It is clear that no node can transmit and receive at the same time since there is no link in the delay graph from a node's receiving node to its transmitting node that has the same slot. A node will not be scheduled to transmit different packets to different receivers at the same time, otherwise it will violate the node-disjoint rule. A node can also receive a packet from a single sender at the one time because of the node-disjoint rule.

Finally, the sum of the weights of the $M$ node-disjoint paths can be interpreted as the total delay of $M$ paths directly. Thus the joint scheduling and routing solution is optimal in terms of latency. ■

References [81, 84] propose algorithms to find $M$ node-disjoint paths with minimum total weights (here denotes delay) is minimized. We can apply the algorithms directly on the delay graph. We first

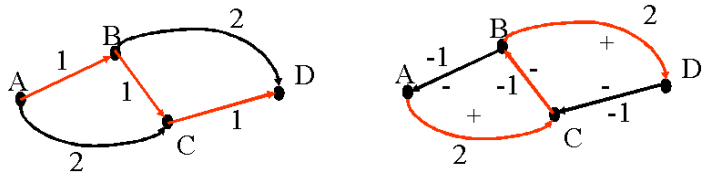



Figure 5.6: An example of finding minimum length 2 node disjoint paths.

briefly describe the algorithm. Figure 5.6 shows an example. We can simply use any shortest path algorithm to find the first minimum weight node disjoint path from $a$ to $z$, where $a$ is the super pseudo source node, and $z$ is the pseudo sink node. We assume $P_M$ is a given optimal set of M node-disjoint paths in delay graph $DG$. Now we need to find optimal $(M+1)th$ node-disjoint paths $P_{M+1}$, as follows:

1. Reverse the direction of each edge on $P_M$, and make its length negative. These edges are called negative arcs. Other edges are called positive arcs.

2. Split each vertex $v$ on $P_M$ into two nodes $v_1$ and $v_2$, joined by an arc of length zero, directed towards $a$. Assign output links on $v_2$ and input links on $v_1$.

3. Find a shortest path from $a$ to $z$ on this transformed delay graph. We call this path an interlacing $S$, which may contain both positive arcs and negative arcs.

4. let $P_M + S$ represent the graph obtained by adding to $P_M$ the positive arcs of $S$, and removing from $P_M$ the negative arcs of $S$. This is called Augmentation, which results in the optimal $P_{M+1}$ paths, the minimum weight $M+1$ node-disjoint paths.

We present a lemma that will be used in the following proofs.

***Lemma** 3:* The output of the augmentation of $P_M + S$ is a set of minimum weight $M+1$ nod-disjoint paths: $P_{M+1} = P_M + S$.

We omit proofs here. Interested readers could see [81, 84] for details.

The authors [81, 84] only proposed algorithm to compute $P_{M+1}$ from $P_M$. This can be extended to compute $M-1$ node-disjoint paths from $M$ node-disjoint path as follows:

1. Reverse the direction of each edge on $P_M$, and make its length negative. These edges are called negative arcs. Other edges are called positive arcs.

2. Split each vertex $v$ on $P_M$ into two nodes $v_1$ and $v_2$, joined by an arc of length zero, directed towards $a$. Assign output links on $v_2$ and input links on $v_1$.

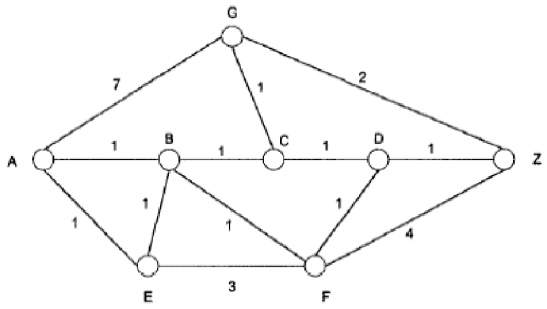



Figure 5.7: An example network with original arc-length

3. Find a shortest path from $z$ to $z$ on this transformed delay graph. We call this path an interlacing $rS$, which may contain both positive arcs and negative arcs.

4. let $P_M + rS$ represent the graph obtained by adding to $P_M$ the positive arcs of $S$, and removing from $P_M$ the negative arcs of $S$. This is called Augmentation, which results in the optimal $P_{M-1}$ paths, the minimum weight $M-1$ node-disjoint paths.

***Lemma 4:*** The output of the augmentation of $P_M + rS$ is a set of minimum weight $M-1$ node-disjoint paths: $P_{M-1} = P_M + rS$.

*Proof:* Suppose we compute $P_M$ from $P_{M-1}$ by augmentation of an interlacing $S$. Suppose the original graph is $G$. According to [81], in an equivalent graph $G'$ of $G$, $P_{M-1}$ contains all negative links of $G'$ and all links on $S$ have 0 length. In the link reversed graph of $G'$, $G'_M$, the exact reverse of $S$, $rS$ have the smallest weight of 0. By augmentation of $P_M$ and $rS$, we will get $P_{M-1}$. It is possible there are other reverse interlacing with length 0, but the result $P_{M-1}$ has the same total length. Thus the augmentation of $P_M + rS$ is a set of minimum weight $M-1$ node-disjoint paths. ■

### 5.3.3 MLSR under Traffic change

The algorithm in reference [81] is aimed to find $M$ node-disjoint path from the beginning. In a real sensor network application, however, flows can appear or disappear dynamically and randomly. The overhead to re-run the whole algorithm whenever there is a change of existing flows is high and also

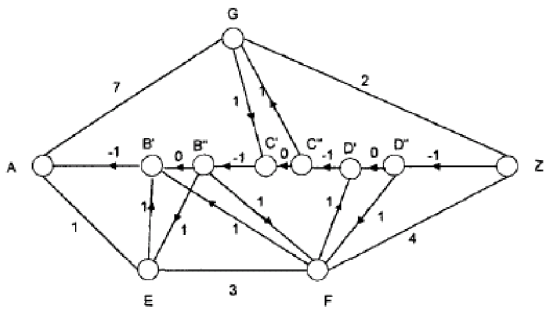



Figure 5.8: An example network after node-splitting and link reverse.

hard to implement. We propose an extension of the algorithm that is able to work on the newly added flow or removed flow.

#### 5.3.3.1 Adding a Flow

Given a Delay Graph and $M$ existing flows, suppose we already have minimum latency $M$ node-disjoint paths. For each source added, we add a pseudo source node in the Delay Graph and connect the super pseudo source node to it. Then we try to construct an interlacing on the new Delay Graph and interpreted it to construct $M + 1$ node-disjoint paths. The paths found maybe different from the paths found by rerun the whole algorithm on the Delay Graph from beginning, but the latencies are same.

In general, the new source could be a new source node or an additional report slot request from an existing source node. Without loss of generality, suppose there is a new source node $A$ that needs to report to sinks.

**AddFlow Algorithm:**

1. Add a pseudo source node $PA$ for source node $A$. Create links from $PA$ to each receiving node or packet generated node of $A$, $R_A^i$ with link delay 1. Also create a link from super pseudo source node $PS$ to $PA$.

2. Assign the link $(PS, PA)$ a very large weight which assure that any path use this link will have longer delay than the path without using it.

3. Find a minimum length interlacing $S$, from $PS$ to $PZ$.

4. By $P_M + S$, we get $P_{M+1}$ minimum latency paths.

*Proposition 5:* AddFlow algorithm reserves optimality.

*Proof:* We will prove it by induction. Suppose we have M optimal paths $P_M$. Now a new source node $S_i$ start to generate packets. We add a pseudo source node $PS_i$ and the necessary links as described in the algorithm. We call this delay graph $G$. Then we assign a very large weight $\beta$ to the link of $(PS, PS_i)$. We call the new graph $G'$. As $\beta$ is large enough that the path using this link always has length larger than any other path. Thus $P_M$ remains optimal in $G'$. Therefore, we can still use algorithm in [81] to get $P_{M+1}$ in $G'$ from $P_M$. Now we want to prove that $P_{M+1}$ is also optimal in $G$.

First the links from $PS$ to all other pseudo source nodes $PS_i$ must be in $P_{M+1}$ as those nodes are the only $M + 1$ out neighbors of $PS$. Second, let us consider only the paths from all $PS_i$ to pseudo sink node $PZ$. These paths are optimal in $G'$ and remain optimal in $G$. Therefore the $M + 1$ optimal paths in $G'$ are still optimal in $G$.

This completes the proof. ■

#### 5.3.3.2 Removing a Flow

Removing a flow from the $M$ flows is different. We will construct a shortest length interlacing that originated from the pseudo sink node to the super pseudo source node. We remove all links from super pseudo source node to the pseudo source nodes, except the one that we want to remove. After interpret the interlacing with the existing $k$ node-disjoint paths, only $K - 1$ paths remains, which is a minimum latency $k - 1$ node-disjoint paths. Suppose we need to remove the flow originated from source node $i$; its pseudo source node in delay graph is $s_i$.

**RemoveFlow Algorithm:**

1. Assign the link from $PS$ to other pseudo source nodes except $PS_i$ a very large weight $\beta$ such that any path using $(PS, PS_i)$ is smaller than the paths.

2. Reverse the direction of each edge on $P_M$, and make its length negative. These edges are called negative arcs. Other edges are called positive arcs.

3. Split each vertex $v$ on $P_M$ into two nodes $v_1$ and $v_2$, joined by an arc of length zero, directed towards $PS$. Assign output links on $v_2$ and input links on $v_1$.

4. Find a shortest path from $PZ$ to $PS$ on this transformed delay graph, which is a reverse interlacing $rS$.

5. let $P_M + rS$ represent the graph obtained by adding to $P_M$ the positive arcs of $rS$, and removing from $P_M$ the negative arcs of $rS$. This is called Augmentation, which results in the optimal $P_{M-1}$ paths, the minimum weight $M-1$ node-disjoint paths.

*Proposition 6:* The output of the augmentation of $P_M + rS$ is a set of minimum weight $M-1$ nod-disjoint paths.

*Proof:* Similarly we can prove the algorithm by induction. It is clear that the output of the augmentation is a set of $M-1$ node-disjoint path. So we only need to prove that it has the minimum weight. let $W_{P_M}$ be the sum of the weights of all the links on $P_M$ and $W_S$ be the sum of the weights of all the links on $S$. The negative arcs on $S$ will be removed from $P_M + S$, with negative length. The same arc (with reverse direction) is on $P_M$ with positive length. The sum of these two links is zero. Positive arcs on $S$ will be added from $P_M + S$, which is not in $P_M$. Thus it is easy to get that $W_{P_M+S} = W_{P_M} + W_S$. Since $S$ is the shortest path from $PS$ to $PZ$ on the transformed delay graph, $W_{P_M+S}$ has the minimum weight. Thus the augmentation of $P_M + S$ is a set of minimum weight $M+1$ node-disjoint paths. It is also clear that $rS$ must use the link $(PS_i, PS)$, thus the flow will be removed from the paths.

■

### 5.3.4 MLSR under Topology change

The basic algorithm in the previous section assumes that the network topology is static while the $k$ disjoint paths are discovered. In this section, we generalize this approach to a network that is undergoing constant topological changes. For instance, a mobile node can simply walk away from the communication network or a link is under strong interference in a harsh environment. We model such changes by link failures. Note that node failures can be modeled as a special case of a certain set of link failures. That is, if node $v$ were to fail, this event can be modeled as the failure of all links adjacent to $v$. Besides link failures, new sensor nodes may be put into the field when there are not enough old nodes available for the application. Or an environmental noise disappeared so a link between two nodes is now available. We model such changes by link joins.

When link failures or link joins happen, the topology of the network changed. If a failed link is currently used by the one of the $k$ minimum latency disjoint paths, the flow using this path is unable to transmit the report. If a new link joins, the existing $k$ disjoint paths may no longer have the minimum latency. Under both situations, a new set of $k$ minimum latency disjoint paths need to be computed. One way is to recompute the $k$ disjoint paths from beginning on the new topology. However, the overhead

is high if topology changes happen frequently. Same as the flow changes, we propose algorithms which can adjust the existing $k$ disjoint paths incrementally with only two computation of the shortest path algorithm.

When a link $e = (u, v)$ currently used by one of the $P_M$ path failed, a new set of $P_M$ is needed. Assume $e$ is used by source $s_i$ and the path is $pi$. The idea is to first remove the flow 1 to get minimum latency $M - 1$ node disjoint path $P_{M-1}$ without using the failed link, then construct a new set of $P_M$ paths. The steps of the algorithm are following:

**LinkFailure Algorithm:**

1. Reverse the direction of each edge on $P_M$, and make its length negative. These edges are called negative arcs. Other edges are called positive arcs. Failed link $e = (u, v)$ now becomes $e' = (v, u)$.

2. Split each vertex $v$ on $P_M$ into two nodes $v_1$ and $v_2$, joined by an arc of length zero, directed towards $PS$. Assign output links on $v_2$ and input links on $v_1$.

3. Find a shortest path from $PZ$ to $v$ on this transformed delay graph. This is a reverse interlacing $rS_1$.

4. Find a shortest path from $u$ to $PS$ on this transformed delay graph. This is a reverse interlacing $rS_2$.

5. let $P_M + rS_1 + rS_2 + (v, u)$ represent the graph obtained by adding to $P_M$ the positive arcs of $rS$, and removing from $P_M$ the negative arcs of $rS$. This is called Augmentation, which results in the optimal $P_{M-1}$ paths, the minimum weight $M - 1$ node-disjoint paths with link $e = (u, v)$ removed.

6. Using algorithm described in section "adding a flow" to construct a new set of $P_M$ minimum latency node-disjoint path.

For LinkJoin Algorithm, suppose the newly joint link is $e = (u, v)$.

**LinkJoin Algorithm:**

1. Reverse the direction of each edge on $P_M$, and make its length negative. These edges are called negative arcs. Other edges are called positive arcs. The newly added link is $(u, v)$.

2. Split each vertex $v$ on $P_M$ into two nodes $v_1$ and $v_2$, joined by an arc of length zero, directed towards $PS$. Assign output links on $v_2$ and input links on $v_1$.

3. Find a shortest path from $PZ$ to $u$ on this transformed delay graph. This is a reverse interlacing $rS_1$.

4. Find a shortest path from $v$ to $PS$ on this transformed delay graph. This is a reverse interlacing $rS_2$.

5. Let $P_M + S_1 + S_2 + (u, v)$ represent the graph obtained by adding to $P_M$ the positive arcs of $S$, and removing from $P_M$ the negative arcs of $S$. This is called Augmentation, which results in the optimal $P'_{M-1}$ paths with $(u, v)$, the minimum weight $M + 1$ node-disjoint paths with link $e = (u, v)$ added.

6. Compute $P''_{M-1}$ by RemovFlow algorithm without link $(u, v)$.

7. Use the smaller of $P''_{M-1}$ or $P'_{M-1}$ as $P_{M-1}$ to compute $P_M$.

### 5.3.5 Other issues

#### 5.3.5.1 Energy efficiency

1. **Energy efficiency:** The energy savings are achieved by the schedule of the nodes; that is, nodes are on only when necessary to deliver the packet and asleep otherwise. However, since our algorithm aims to minimize the latency, the route may take longer hops than a minimum hop routing algorithm. However, the energy saving by the schedule is more significant than the possible energy cost by extra number of hops.

2. **Load balancing:** Although our algorithm does not take load balancing into direct consideration, the algorithm will naturally achieve load balancing since a heavy loaded node is more likely to have high delivery latency, and is hence avoided by the algorithm.

#### 5.3.5.2 Distributed solution

The algorithm can be easily implemented in a distributed manner. The only operation needed in the algorithm is to find the shortest path between source and sink, with possible negative weights (no negative cycles). Distributed versions of Bellman-Ford algorithm can be employed directly [89].

Bellman-Ford algorithm has been used widely in Internet, known as the Distance Vector routing algorithm specified in [86]. In the decentralized implementation in Internet, each node periodically broadcasts its routing tables, which contains the cost from itself to all other nodes, to all its neighbors. A router updates its own routing table according to the routing tables of its neighbors. The algorithm

will converge to the optimal solution after certain number of iterations. Each node then knows the next hop to forward a packet by its routing table. AODV [88], Ad hoc On demand Distance Vector, is an on-demand routing protocol for wireless multi-hop networks, that is able to find the shortest path between two nodes based on Bellman-Ford algorithm. Thus in this work, we assume that distributed version of shortest path algorithm is available. Since we assume that traffic flows are long lived, the overhead of distributed Bellman-Ford algorithm can be ignored.

### 5.3.6 Heuristic solutions

We will compare the performance of our algorithms with the following two simple heuristic solutions.

1. **Naive Dijkstra algorithm:** This algorithm is a very basic algorithm that finds the link disjoint paths. It entails running Dijkstra's shortest path algorithm k times on the Delay Graph G, where after each run, links belongs to the last path found are removed, ensuring link-disjointness among the $k$ paths. A new Dijkstra's shortest path is found each time a new flow is added. When a flow ends, links belonging to the path of the flow is recovered. When a link is broken, if it is used by one of the path, the path will be removed and then a new path is computed for that source. Nothing is done if a new link is added. We will refer this algorithm as $NaiveD$.

2. **Centralized Dijkstra algorithm:** This algorithm is different from the previous algorithm in that the algorithm is run on the Delay Graph G that take all the existing flows into consideration. Each time a new flow is injected or an existing flow is removed, the delay graph G is reconstructed, then the Dijkstra algorithm is run $k$ times to find $k$ node-disjoint path, with links belongs to the previous path found removed. We will refer this algorithm as $CentraD$.

## 5.4 MLSR under Interference

In last section, we assume that links in interference range can use different channels. In this section, we assume that all nodes share the same wireless channel, thus interference need to be taken care of. We assume a disk interference mode, in which a sender node will interfere with another communication link if and only if it is among certain distance from the receiver of that link.

Clearly this problem is NP-hard (since the TDMA scheduling problem is NP-hard), so we proposed a heuristic approach. We define paths to be interference-free if there are no interference among the links on the paths. First we shown how to find the first interference path for a flow:

1. Find the shortest delay path for the flow on the delay Graph $DG$.

2. If there is no interference among links on the path, then finished. Otherwise, remove the link that causes the most interferences. Check if there is still interference. Keep deleting links that cause the most interference until there is no interference in the path. Go back to step 1.

Now suppose we have $M$ node-disjoint interference-free paths, we need to find $M+1$ node-disjoint interference free paths.

1. Remove links in $DG$ that interfere with the links on the $M$ node-disjoint paths.

2. On the remaining $DG$, find a minimum latency interlacing $S$ using algorithm described in [81]. If there is no interference among links in $S$, go to step 4. Otherwise go to step 3.

3. Remove the link that causes the most interferences, which will be among the new links introduced by $S$. Check if there is still interference. Keep removing links on the paths that cause the most interference until there is no interference in the paths (the links in the original $M$ node-disjoint interference free paths won't be removed). Go back to step 2.

4. Get the $M+1$ node-disjoint interference free path by interpret $P_M + S$. Recover the previously removed links that interfere with links that was in $M$ node-disjoint paths but not in $M+1$ node-disjoint paths. Remove links that interfere with the newly added links on the $M+1$ node-disjoint interference free paths.

## 5.5 Numerical Results

In this section, we evaluate the performance of the MLSR algorithm and heuristic algorithms through high level simulations. Since the current study focuses on comparing the total latency only across these heuristics, even the distributed algorithms are simulated in a centralized manner (without analyzing their overhead).

A sensor network is generated by randomly scattering 200 nodes on a 200x100 rectangle. There are four sinks in the corners of the area. The radio range is set to be 10 meters. We use the topology generation tool provided by [68] to get the packet reception ratio (PRR) of links between two nodes. Links with PRR larger than 0.9 are added into the abstract communication graph.

### 5.5.1 FDMA channel Model

First we evaluate the algorithm under the FDMA model which means there is no interference between two links. Figure 5.9 shows the average delay under different frame length for MLSR, $NaiveD$ and $CentraD$ algorithms. For each frame length $k$, the number of flows are $4K$ which is the largest possible number of flows that can be supported by the 4 sinks. The results are averaged over 10 different seeds. Clearly MLSR has the smallest average latency while $NaiveD$ has the largest latency. Compared to $NaiveD$ algorithm, the delay of MLSR is reduced to around 15%. As $K$ increases, the average latency decreases. This is because as number of flows increases, more flows are closed to the sink, thus the latency is reduced.

Figure 5.11 shows the $NaiveD$ algorithm under different flow join order. In this scenario, $K = 5$, the number of flows $FN = 20$. The flows are fixed but the order of being processed by the $NaiveD$ is different. Clearly the order will affect the performance of the algorithm. However, $CentraD$ algorithm is likely to achieve lower latency while MLSR achieves the minimum latency regardless of the order.

Figure 5.12 shows the performance of MLSR and $NaiveD$ under traffic changes: flows comes in order and then later leaves. The $CentraD$ algorithm is not simulated here as it can not handle traffic change adaptively. For the first 27 flows, the $NaiveD$ algorithm achieves the same latency as MLSR. This shows that when the number of flows is small, there is no need to do the complicated MLSR algorithm. However when the number of flows is larger than 27, the MLSR has smaller latency. Later when flows finished their data transmission in the same order as they joins (the first joined flow first left), the MLSR always achieves the minimum latency for current active flows. The naive algorithm, however, performs poorly even when there are less then 27 active flows, it has higher latency than MLSR.

Figure 5.13 and 5.14 show the performance of MLSR and $NaiveD$ under topology change with $k = 10$. We keep removing links and computes the latency after each link failure until we can not find 30 or 40 node-disjoint paths. As link failure happens, the latency increases. The MLSR achieves smaller average latency under various topology changes than $NaiveD$.

### 5.5.2 Single Channel Interference

In this section, we investigate the performance of the heuristic MLSR algorithm under single channel interference model through simulations. The interference range is set to 20 meters. Figure 5.15 shows the number of average delay of flow under different $K$ and $FN$. As $K$ increases, the number of flows can be supported also increase. For each $K$, there is a big jump of latency at certain point. Since each

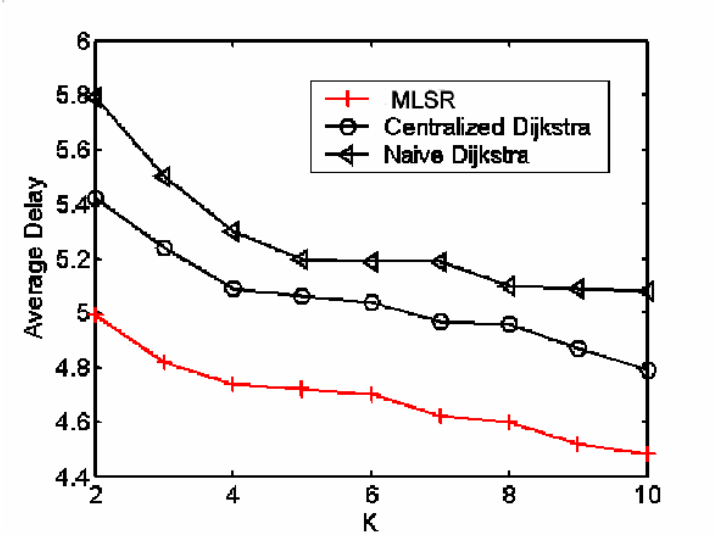



Figure 5.9: Average delay under different frame length

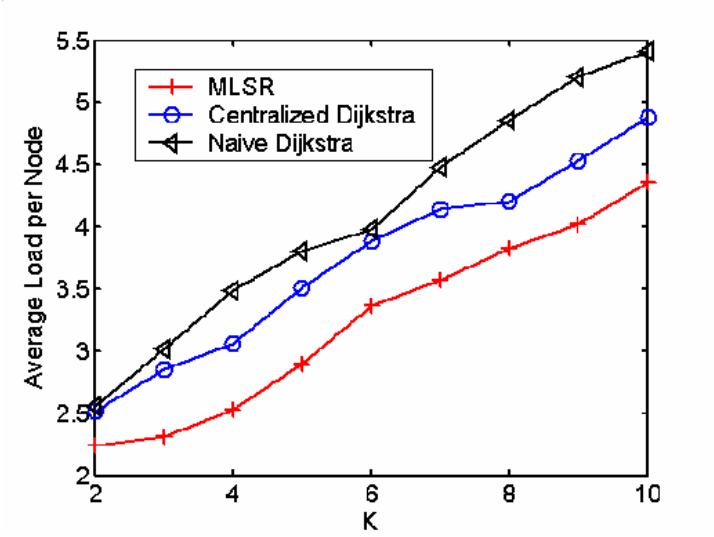



Figure 5.10: Average load per active node under different frame length

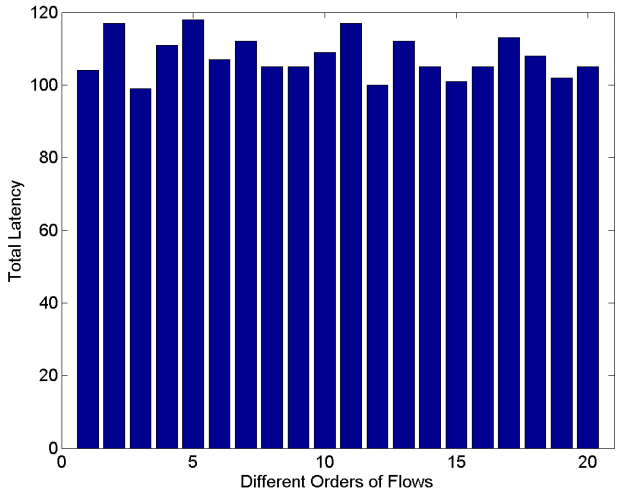



Figure 5.11: Total Delay of $NaiveD$ heuristic under different flow join order

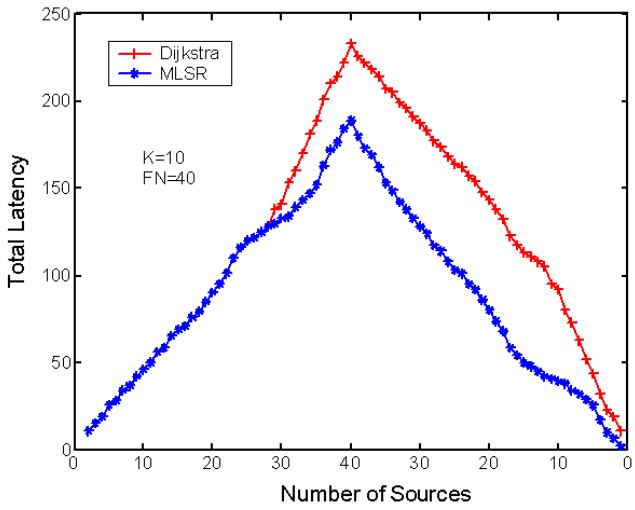



Figure 5.12: Total delay of $NaiveD$ heuristic and MLSR for flow joins and leaves

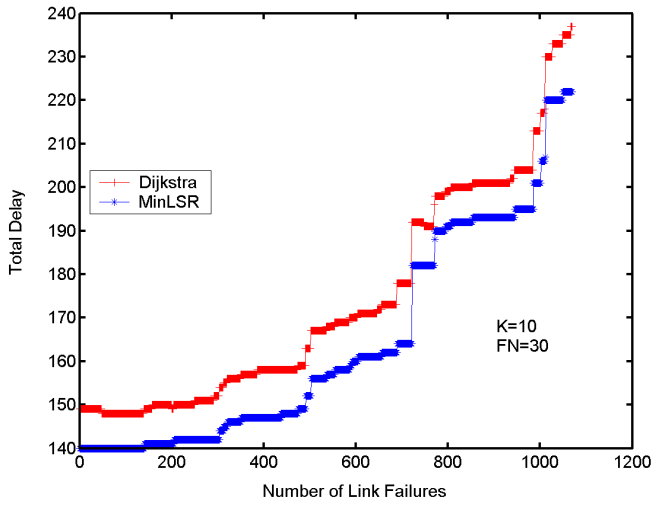



Figure 5.13: Total delay of $NaiveD$ heuristic and MLSR under topology change

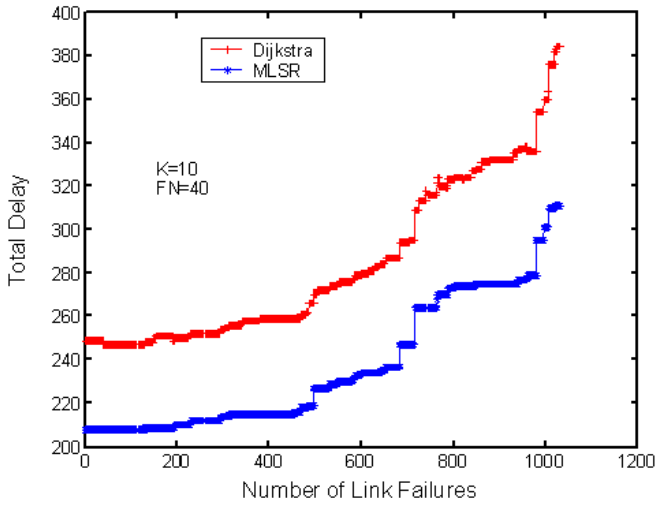



Figure 5.14: Total delay of $NaiveD$ heuristic and MLSR under topology change

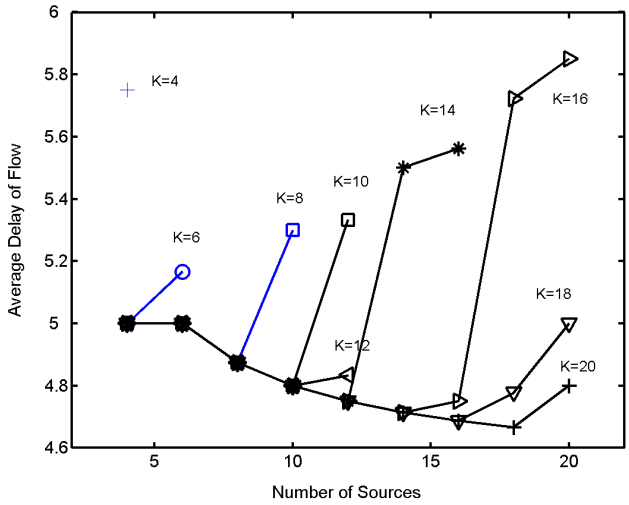



Figure 5.15: Average Delay under different frame length and number of flows

source has a packet to sent every $K$ slots, this means the traffic load is too heavy for current network to support.

## 5.6 Discussion

In this chapter we addressed the important problem of minimizing communication latency while providing energy-efficiency for nodes in wireless sensor networks. Different from DESS whose objective is to minimize the worst case latency given the duty cycling requirement that each sensor has to be awake for $\frac{1}{k}$ fraction of time slots on an average, MLSR is interested in the average latency for the current active flows. A node is allowed to wake up multiple slots to receive/transmit and asleep otherwise. We formulated a joint scheduling and routing problem with objective to find the schedule and route for current active flows with minimum average latency. By constructing a delay graph, the problem can be solved optimally by $M$ node-disjoint paths algorithm under FDMA channel model. We further extended the algorithm to handle dynamic traffic changes and topology changes in wireless sensor networks. We also proposed a heuristic solution for the minimum latency joint scheduling and routing problem under single channel interference. Numerical results show the latency can reduced 15% under stationary scenario and 50% under dynamic traffic or topology changes.

# Chapter 6

# EEJSPC: Energy Efficient Joint Scheduling and Power Control

## 6.1 Overview

In previous chapters, we assumed radio models with fixed transmission power. However, many radios now can support multiple level of transmission power which provide another knobs for system design. In this chapter, we will study the problem of energy efficient joint scheduling and power control.

TDMA scheduled medium access is generally more energy efficient than random access, and is particularly suitable for implementation with low overhead when traffic is predictable or slowly changing. Several studies have investigated TDMA scheduling techniques for ad hoc and sensor networks [6, 7, 8, 12, 9, 10, 13]. In these studies, typically a simple model for interference is used where a receiving node sees interference from another transmitter if and only if it is within some nominal range $R_I$. This model, while useful in providing a simple graph-coloring approach to TDMA scheduling, can be quite misleading in practice. In reality, simultaneous wireless transmissions within the nominal range do not necessarily collide if the signal to interference plus noise ratios (SINR) at the corresponding receivers are sufficiently high; and, at the other extreme, aggregate interference from multiple transmitters that are well beyond the nominal range can be high enough to cause collisions.

Another concern with many studies of TDMA in wireless ad hoc and sensor networks is that they ignore the possibility of variable transmission power. In practical systems this can be an important tunable parameter for reliable and energy-efficient communication, because higher transmit powers can increase the SINR at the receiver to enable successful reception on a link, and lower transmission power can mitigate interference to other simultaneously utilized links.

We treat in this work TDMA link scheduling using a realistic SINR-based interference model, explicitly taking transmission power control into account. This approach to joint scheduling and power control was first taken by ElBatt and Ephremides [15, 16], followed by others including [17, 18, 19, 20,

67]. Given a set of one-hop links and number of packets that need to be transmitted within a certain number of slots, the *scheduling problem* is to decide in each time slot which source-destination pairs communicate while *power control problem* is to decide the transmission power of source nodes in a given slot.

In these prior works, the primary objective of the link scheduling algorithm is to maximize the number of simultaneous transmissions which maximize the throughput. While the power control phase minimizes transmission powers on the scheduled links, link scheduling can not guarantee power efficiency, because maximizing the concurrent transmissions increases inter-sender interference and hence the total required transmission power. Potentially significant energy savings are possible through alternate link schedules. Even further energy savings may be achievable by trading off throughput and latency.

In this chapter, we study the energy efficient joint scheduling and power control problem. Our contributions in this work are four-fold. First, we formulate joint scheduling and power control as a novel optimization problem that provides tunable tradeoffs between throughput, energy and latency. We show that the prior formulations in [15, 17] can in fact be treated as special cases of our formulation. Second, while the optimization problem that we formulate is NP-hard, we present both exponential and polynomial complexity greedy based heuristic algorithms. Third, we show the performance of these algorithms through simulation results and demonstrate the energy-latency-throughput tradeoffs that can be achieved with joint link scheduling and power control. Interestingly, we find that, at least for moderate loads, major energy savings can be obtained without significantly sacrificing throughput. Finally, we study the the energy efficient joint scheduling and power control problem with the objective of minimizing minimize the total energy cost subject to all packets of the links are transmitted within a latency bound.

The rest of the chapter is organized as follows. In section 6.2, we define the energy efficient joint scheduling and power control problem. We study the tunable joint link scheduling and power control problem in section 6.3. In section 6.4, we investigate the problem of joint scheduling and power control with transmission request constraint. We evaluate the performance by simulations in section 6.5. We then summarize and discuss the work in 6.6.

## 6.2 Energy, Latency and Throughput Tradeoffs in JSPC

### 6.2.1 Application Scenario

We first describe the basic application scenario and assumptions.

1. Consider a static wireless sensor network, all nodes are equipped with same radio with omni-directional antennas and share the same channel. The transmission power of the radio can be adjusted continuously [1], with constraints on the minimum and maximum transmission power levels. The radio data rate is fixed.

2. Consider a general application in wireless sensor network, each sensor node samples the environment periodically. A node either reports to sink or communicate with neighbors when an interesting event is detected. Sensing data need to be processed or reported before a latency deadline, such as in fire detection or real time target tracking applications.

3. The deadline can be per-hop deadline or end-to-end deadline. In case of end-to-end deadline, we divide the end-to-end deadline by the number of hops so we have a per-hop deadline for each link on the path. This is reasonable since end-to-end deadline should be proportional to the number of hops on the path.

4. Time is divided into equal sized slots that are long enough for one packet transmission and grouped into frames. Some works on TDMA focus on minimizing the length of the frame subject to the constraint that every node or link is assigned at least one slot. In this work, however, the frame length is chosen according on the per-hop latency deadline.

5. Each node generates random number of packets of fixed length which need to be transmitted in one TDMA frame. This is called a transmission request. Packets not transmitted within the current time frame are dropped.

6. For end-to-end data packet (e.g, from a sensor to the sink), every TDMA frame, it will be forwarded one hop to a neighboring node. In the next TDMA frame, the packet will then be counted as the transmission request of the neigh node until it reaches the sink.

[1] In practice, there may only be several discrete transmission power levels. This assumption, however can simplify the analysis and does not affect the correctness of the algorithm.

### 6.2.2 Interference Model

The interference model that we consider is a SINR-based TDMA system. Let $G = (V, E)$ be the wireless sensor network, with $V$ representing the set of nodes in the network and $E$, the set of communication links. Given a link $(i, j) \in E$, $i$ is the sending node and $j$ is the receiving node. A link is called active in a slot if node $i$ transmits data packet to node $j$ in that slot. We refer all active links in a single time slot as a *transmission scenario*, or *transmission set*. The signal to interference and noise ratio ($SINR$) for link $(i, j)$ is defined as:

$$SINR_{ij} = \frac{\alpha_{ij} P_i}{N_j + \sum_{k \neq i} \alpha_{kj} P_k} \tag{6.1}$$

where $\alpha_{ij}$ is the propagation attenuation of the signal from node $i$ to node $j$, which is proportional to $\frac{1}{d_{ij}^n}$, where $n$ is the path loss factor. We assume $\alpha_{ij}$s changes slowly so that we can regard $\alpha_{ij}$ as constant for the duration of a time frame in the following discussion. $N_j$ is the environment noise power at receiver $j$. $P_i$ and $P_k$ are the transmission powers of sending node $i$ and $k$ separately.

A data transmission on a link $(i, j)$ can be successfully received at the receiver only if the corresponding $SINR$ on that link is equal or greater than a given threshold $\gamma$:

$$SINR_{ij} \geq \gamma \tag{6.2}$$

### 6.2.3 Power Control

If there is only one active link $(i, j)$, node $i$ only needs to transmit at a power level just high enough to satisfy $SINR_{ij} \geq \gamma$. However, if there are multiple active links in the same time slot, because of the interfere among each other each node has to transmit at higher power in order to meet the $SINR \geq \gamma$ requirements, which increases the interference in return. The power control problem is to compute a set of transmission power for all links in a transmission scenario by solving the following optimization problem:

$$\begin{aligned} \text{minimize} \quad & \sum_{ij} P_{ij} \\ \text{subject to} \quad & SINR_{ij} \geq \gamma \\ & P_{min} \leq P_{ij} \leq P_{max}, \ \forall ij \text{ links} \end{aligned} \tag{6.3}$$

Some distributed power control algorithms have been proposed for cellular network [14] and wireless ad hoc networks [15], which we will use directly.

We call a transmission scenario/set *feasible* if a set of transmission powers are available such that the $SINR$ requirements of all receivers in the transmission scenario are satisfied. A set is called a *maximal transmission set* if adding any additional active link will result in an infeasible transmission set. All subsets of a *maximal transmission set* are also *feasible transmission sets*. We refer the sum of the transmission power of all active links in a transmission scenario as its energy cost.

We make two important observations about the total transmission power of a feasible transmission scenario.

1. Two feasible transmission scenarios with same number of concurrent transmissions could have significantly different costs because of the different interference among the the links, depending on the location and wireless channel of the links.

2. A feasible set's cost is always larger or equal to sum of the costs of its subsets.

The first observation needs no further clarification. We use a single example to explain the second observation. Consider a set $S$ and its subset $S - l$ (remove link $l$ from $S$) and $l$. Suppose the costs are $C_{S-l}$ and $C_l$. Now add $l$ to subset $S - l$. $l$'s transmission power will interference with links in $S - l$ and vice versa. Therefore both link $l$ and links in $S - l$ have to increase their transmission power respectively. Clearly, $C_S > C_{S-l} + C_l$. The subsets in the right hands do not need to be exclusive to each other, as the redundant links will only increase the transmission cost.

$$\begin{aligned} \text{If} \quad & S_j = \bigcup_k S_{jk} \\ \text{then} \quad & C_j \geq \sum_k C_{jk} \end{aligned} \tag{6.4}$$

### 6.2.4 State of the Art of Joint Scheduling and Power Control

In previous works on joint scheduling and power control [15, 16, 17, 67], the scheduling policy is to pack the maximum number of links that can be active simultaneously in each time slot. The objective is to maximize the spatial reuse of system resources and the throughput. Although the power control phase minimizes the transmission powers on the scheduled links, this scheduling policy does not take energy into consideration and thus may not be energy efficient.

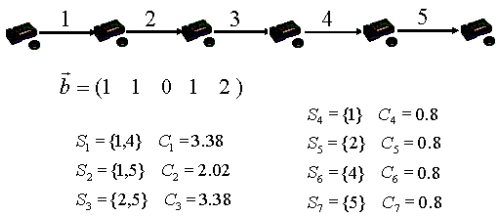



Figure 6.1: Illustration of energy efficient scheduling. $\vec{b}$ is the number packets need to be transmitted for each link. $S$ are all possible feasible transmission scenarios. $C$ are the total transmission power of the transmission scenarios.

Figure 6.1 shows an example of energy efficient joint scheduling and power control[2]. Given $\vec{b}$, the number of packets need to be transmitted and all feasible transmission scenarios and their related costs, there are three possible schedules that satisfy the $\vec{b}$ constraint:

1. Option 1: Choose $S_1$, $S_3$ and $S_7$. The transmission request is finished in three slots. The total energy cost is 7.56.

2. Option 2: Choose $S_2$, $S_3$ and $S_6$. The transmission request is also finished in three slots. The total energy cost is reduced to 6.2.

3. Option 3: Choose $S_2$, $S_5$, $S_6$ and $S_7$. The transmission request now is finished in four slots. The total energy cost is further reduced to 4.42.

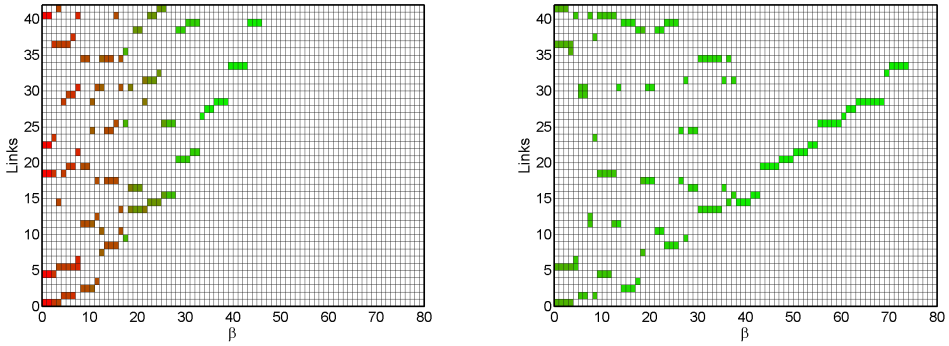
This example shows that **the scheduling policy that maximizes the number of concurrent transmissions is not energy efficient** and suggests two ways to achieve energy efficient schedule:

1. Choose energy efficient combination of feasible transmission sets. In the example, compare option 2 to option 1, the combination of $S_2 + S_6$ is more energy efficient than $S_1 + S_7$. This is because the interference between link 1 and 4 is higher than the interference between link 1 and 5.

2. Tradeoff latency for energy efficiency. In the example, compare option 3 to option 2, $S_3$ is divided into $S_5 + S_7$. Instead of being scheduled simultaneously in one slot, link 2 and 5 are scheduled separately in two slots. Because of the elimination of interference, the total energy cost is further reduced.

To better understand the two approaches to save energy, figure 6.2(a) and 6.2(b) show two different schedules. Each column is a slot and each colored box represents an active link during that slot. The

[2]The data is collected by simulations described in section V




(a) $\beta = 0$ (b) $\beta = 10$

Figure 6.2: An example of two different schedules under $\beta = 0$ and $\beta = 10$.

color of the boxes in a column indicates the the energy cost of the transmission set in that slot. Red color means high energy cost while green color means low energy cost. The meaning of $\beta$ will be explained later. When $\beta = 0$, the transmission request is finished in less than 50 slots and many transmission sets have high energy cost. While in the schedule chosen by $\beta = 10$, the transmission request is finished in more than 70 slots. Even for two sets having the same number of active links, the energy cost of the set chosen by $\beta = 10$ has much lower energy cost compared to the set chosen by $\beta = 0$.

In the following sections, we will investigate two different problems of energy efficient joint scheduling and power control.

## 6.3 TJSPC: Tunable Joint Scheduling and Power Control

### 6.3.1 Mathematical Formulation

In this section, we will formulate the tunable joint scheduling and power control problem and show that prior works [15, 17] can be treated as special cases of our formulation. First we describe the notation used.

Assume that a TDMA time frame contains $T$ slots. Here $T$ models the per-hop delay tolerance of the application. The duration of a slot is normalized to 1. Let $b(e)$ denote the number of packets need to be sent on link $e = (i, j) \in E$ in a time frame. Denote $\vec{b}$ as a vector of size $|E|$ with each element corresponding to a link.

We denote $S$ as the collection of all feasible transmission sets and $|S|$. Each feasible transmission set $S_k$ is a vector of size $E$, with $S_k(e)$ equal to 1 if $e$ is active in the set $S_k$. For each feasible set $S_k$, there is an energy cost $C_k = \sum_{S_k(e)=1}(P_e)$ which is the sum of the energy cost all active links in that

Table 6.1: Summary of the Notations

| | |
|---|---|
| $e = (i, j)$ | A link with $i$ the sender and $j$ the receiver |
| T | Number of slots in a TDMA frame |
| $\vec{b}$ | Transmission request: Number of packets need to be sent for each link |
| S | The collection of all feasible transmission set, $S_k$ being one of the set |
| $S_k(e)$ | 1 if link $e$ is active in transmission set $S_k$ |
| M | $\|S\|$, number of feasible transmission set |
| $C_k$ | Energy cost of transmission set $S_k$ |
| $\vec{x}$ | The scheduling solution: $x_k$ is the number of times $S_k$ is chosen |
| $\beta$ | Parameter to tune between throughput and energy |

set in a single slot. Here, $P_e$ is the transmission power of link $e$ from $i$ to $j$. We ignore the reception power as it is almost constant regardless of the transmission power. Let $\vec{x}$ denote the solution, with $x_k$ being the number of times that set $S_k$ is chosen. The maximum number of sets allowed to be chosen is $T$.

The three important metrics of a sensor network system can be easily represented using the above parameters.

- **Energy**: The total communication energy cost in $T$ slots: $\sum_k x_k C_k$.
- **Throughput**: We use the number of packets transmitted in $T$ slots to represent the throughput. The number of slots that a link $e$ is scheduled to be active is $\sum_k x_k S_k(e)$. However if a link is assigned a slot but there is no more packet to transmit, it is a waste of resource and should not be counted. So the actual number of packet a link $e$ transmits is $min(\sum_k x_k S_k(e), \vec{b}(e))$ The total number of packets transmitted by all links is then: $\sum_{e \in E} min(\sum_k x_k S_k(e), \vec{b}(e))$.
- **Latency**: $T$ is the worst per-hop latency of a packet if it is transmitted. A smaller $T$ means that a packet need to be transmitted in a shorter time frame, and hence a smaller per-hop delay.

It is clear that it is not possible to optimize these three metrics simultaneously. Depending on the application requirements, different tradeoff strategies may be used. Some applications may need all transmission requests be satisfied before the deadline, while others may tolerate a certain number of packet drops. We will study the energy cost minimizing problem subject to transmission request guarantee in section IV. In this section we first form a problem that allows the applications to choose different tradeoffs among energy, latency and throughput.

Problem TJSPC:

$$\begin{aligned} \text{max } gain &= \alpha \sum_{e \in E} min(\sum_{k} x_k S_k(e), \vec{b}(e)) \\ &\quad -\beta \sum_{k} x_k C_k \\ \text{s.t. } \sum_{k} x_k &\leq T \end{aligned} \tag{6.5}$$

By tuning $\alpha$, $\beta$ and $T$, we can achieve different tradeoffs between throughput and energy given the latency constraint. Specially, if $\alpha$ is 0, the problem is reduced to minimizing energy consumption with no constraint on throughput. Then the policy of the scheduling algorithm is to always search the set with minimum energy cost. If $\beta$ is 0, the problem is reduced to maximizing throughput with no constraint on energy consumption. Then the objective of the scheduling algorithm is to maximize the throughput, same as previous scheduling algorithms[15, 17]. Without loss of generality, we will assume $\alpha = 1$ in the following discussion. As $\beta$ increases, to maximize the gain it is better to choose transmission scenario with less energy cost. So the application can increase $\beta$ when it is more interested in saving energy and decrease $\beta$ when the throughput is a more important metric. The choice of $T$ would be based on application-specific worst hop-to-hop latency requirements.

As $\beta$ increases, the solution tends to choose transmission sets with smaller energy cost. However, to prevent a transmission set from being chosen because of its low energy cost even if it does not contribute any throughput, there should be an upper bound for $\beta$. Let $C_{min} = \min_k C_k$ and $C_{max} = \max_{k,|S_k|=1} C_k$. It is easy to see that to guarantee that a transmission set that can at least contribute 1 to the throughput is preferable to the set with minimum cost, we have:

$$-C_{min} < 1 - \beta C_{max} \Rightarrow \beta < \frac{1}{C_{max} - C_{min}} \tag{6.6}$$

This problem is NP-hard as it can be reduced from the Maximum Coverage problem [23]. However based on the fast greedy heuristic algorithm with constant factor approximation in [23], we propose greedy based heuristic algorithms and evaluate the performance by simulations.

### 6.3.2 Heuristic Approaches

#### 6.3.2.1 Exponential Complexity Greedy Approximation

In this section, we present a greedy algorithm that has a constant factor approximation to the optimum solution. Given the collection of all feasible transmission sets $S$, the greedy algorithm selects $T$ transmission sets by iteratively choosing the set that maximize the total $gain$ (defined in problem TJSPC) of the already chosen sets plus the current chosen set. We denote this algorithm as Greedy.

The greedy heuristic can be proved to be a $(1 - \frac{1}{e})$-approximation algorithm.

*Proposition 7:* $wt(\vec{x}) \geq [1 - (1 - \frac{1}{k})^k]wt(OPT) > (1 - \frac{1}{e})wt(OPT)$

This follows from the lemma 3.13 in [23]. For completeness, we show the proof here.

*Proof:* Suppose $\vec{x}^i$ is the greedy solution in the first $i$ slots, let

$$G_i = \sum_k x_k^i S_k$$

and

$$wt(G_i) = \sum_e min(\sum_k x_k^i S_k, \vec{b}(e)) - \beta \sum_k x_k^i c_k$$

Suppose in $i+1$ slot, transmission set $S_j$ is chosen, Then $x_j^{i+1} = x_j^i + 1$ and $G_{i+1} = G_i \cup S_j$, we have:

$$wt(G_{i+1}) = \sum_e min(\sum_k x_k^{i+1} S_k, \vec{b}(e)) - \beta \sum_k x_k^{i+1} c_k$$

Now, the gain of the first selected $(i-1)$ sets is $wt(G_{i-1})$. The difference between $wt(G_{i-1})$ to the gain of the optimal solution is $wt(OPT) - wt(G_{i-1})$. Then at least $wt(OPT) - wt(G_{i-1})$ worth of gain not covered by the first $(i-1)$ sets are covered by the $T$ sets of $OPT$. By the pigeonhole principle, one of the $T$ sets in the optimal solution must cover at least $\frac{wt(OPT)-wt(G_{i-1})}{T}$ worth of gain. Since $S_j$ is a set that achieves maximum additional gain, it must also cover at least $\frac{wt(OPT)-wt(G_{i-1})}{T}$. That is:

$$wt(G_i) - wt(G_{i-1}) \geq \frac{wt(OPT) - wt(G_{i-1})}{T}$$

Now let us suppose for $i = 1$, $wt(G_1) \geq \frac{wt(opt)}{T}$, then,

$$\begin{aligned} wt(G_{i+1}) &= wt(G_i) + (wt(G_{i+1}) - wt(G_i)) \\ &\geq wt(G_i) + \frac{wt(OPT) - wt(G_i)}{T} \\ &= (1 - \frac{1}{T})wt(G_i) + \frac{wt(OPT)}{T} \\ &\geq (1 - \frac{1}{T})(1 - (1 - \frac{1}{T})^i)wt(OPT) + \frac{wt(OPT)}{T} \\ &= (1 - (1 - \frac{1}{T})^{i+1})wt(OPT) \\ &> (1 - \frac{1}{e})wt(OPT) \end{aligned}$$

■

When $\beta$ is 0, to maximize the gain, the greedy algorithm will choose a feasible transmission set which can maximize the throughput, which leads to the solution to choose the set with maximum concurrent transmission. This is exactly the scheduling algorithm in [15, 17].

The complexity of the greedy algorithm is upper-bounded by $O(T|S|)$. A loose upper bound on $|S|$ is $2^E$, which means that the complexity of the algorithm is exponential to the number of links. With the feasibility constraint, $|S|$ can be greatly reduced. Cluster hierarchical structures which have been proposed widely for wireless sensor networks (e.g. in [21, 22]) can further reduce $|S|$. Since cluster size are chosen to accommodate event monitor range, it is expected that at any time if an event happens, most of the time only one cluster may need to be active. Each cluster only schedules its own data transmission while treating interference from other clusters as ambient noise. Interference from clusters far away is negligible. Because only links within one cluster need to be considered, the number of feasible transmission sets is reduced considerably. We can further limit the maximum number of concurrent transmission links to a small number $k$, since in practice as it is difficult to sustain a large number of simultaneously active links in a given region. In this case, the number of feasible sets is upper bounded by $2^{k+1}$.

Even $|S|$ can be reduced, the greedy algorithm needs to compute all possible transmission sets $S$ and their energy cost in advance and has an exponential complexity of $O(T|S|)$ whenever the wireless channel condition changes, which makes it infeasible for practical use. However, it could be used as a framework or offline algorithm to give good insight on the performance of the network. In the next section, based on the greedy approximation algorithm, we propose a greedy based heuristic which does not need to pre-compute all feasible transmission sets with polynomial complexity.

#### 6.3.2.2 Polynomial Greedy Heuristic

Assume that the link gain $\alpha_{ij}$ changes slowly compared to time frame $T$, the nodes need only to collect such information until a significant change of $\alpha_{ij}$ happens. The parameters can also be updated incrementally. Therefore, we assume $\alpha_{ij}$ is available in each node. Secondly, at the beginning of each time frame, source nodes will generate a control packet that contains the number of packets intended to its receivers. Therefore all source nodes are aware of $\vec{b}$. We assume the control packet is smaller compared to the data packet and the overhead is small. We will not discuss the details of the control message exchange protocol here.

Given a transmission scenario, a source node first check whether it is feasible. If it is infeasible, a link with minimum $SNR$ or *Maximum Interference to Minimum Signal Ratio(MIMSR)* [17] is deferred. Then the new transmission scenario is checked again. Previous scheduling algorithms will stop once an feasible transmission set is found. The proposed algorithm, however, continues to search for a transmission set that can maximize the gain. Suppose the first admissible set is $S_k$, it will continue to drop the link with maximum MIMSR until there is only one active link. Suppose the following transmission sets the node gets are $S_{k1}, S_{k2}, ..., S_{kn}$. It is clear all these transmission sets are still feasible and $S_k \supset S_{k1} ... \supset S_{kn}$. For each feasible set $S_{ki}$, the node computes the related gains by $\alpha \sum_{e \in E} min(\sum_k x_k S_k(e), \vec{b}(e)) - \beta \sum_k x_k C_k$. Then the transmission set with the maximum gain is chosen and $\vec{b}$ is updated. The whole process is repeated again until either $\vec{b} = 0$ or $T$ sets are chosen. We denotes the algorithm as DiGreedy.

**Algorithm** *DiGreedy*

1. Collect $\vec{b}$.
2. **for** $i \leftarrow 1$ **to** $T$
3. $m \leftarrow$number of unzero element in $\vec{b}$
4. $S(e) \leftarrow 1$ if $b(e) \geq 1$
5. **for** $j \leftarrow m$ **to** $1$
6. Run power control algorithm for $S$
7. $gain \leftarrow \alpha \sum_{e \in E} min(\sum_k x_k S_k(e), \vec{b}(e))$
8. $-\beta \sum_k x_k C_k$ if $S$ is feasible
9. defer the link $k$ with MIMSR
10. $S(k) \leftarrow 0$
11. Select the feasible transmission set $S$ with maximum $gain$
12. $\vec{b} \leftarrow (\vec{b} - S)$

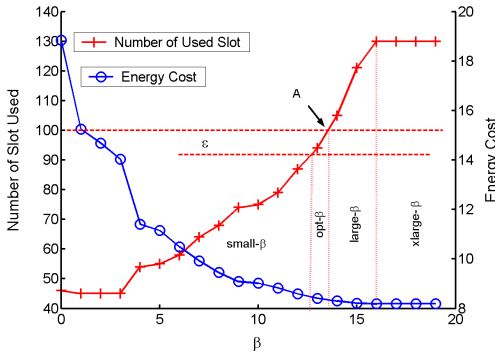



Figure 6.3: The four characteristic regions in the number of used slot, energy vs. $\beta$.

13. **if** $\vec{b} == \vec{0}$

14. break;

The proposed DiGreedy algorithm has a complexity of $O(T|E|)$ which is polynomial to the number of links. However, unlike the greedy algorithm which always choose the transmission scenario that maximizes $\alpha \sum_{e \in E} min(\sum_k x_k S_k(e), \vec{b}(e)) - \beta \sum_k x_k C_k$ from all possible transmission sets, the DiGreedy algorithm only choose one from the transmission sets that are obtained by deferring the MIMSR link one by one. Therefore, it does not necessarily guarantee a $(1 - \frac{1}{e})$-approximation to the optimal solution. However it is practically implementable and we will show by simulations that it achieves comparable performance to the greedy algorithm.

## 6.4 JSPC-TR: JSPC with Transmission Request Constraint

### 6.4.1 Problem Formulation

In TJSPC, we investigate the tradeoffs between throughput and energy efficiency. However, some applications may require all the transmission requests be satisfied. So in this section, we study the problem of joint scheduling and power control with transmission request constraint (JSPC-TR): given a transmission request, minimize the energy cost subject to the constraint that all transmission requests are satisfied within the latency bound:

Problem JSPC-TR:

$$\begin{aligned} \min \quad & \textstyle\sum_k x_k C_k \\ \text{s.t.} \quad & \textstyle\sum_k x_k S_k(e) \geq b(e),\ \forall e \\ & \textstyle\sum_k x_k \leq T \end{aligned} \tag{6.7}$$

This is still a NP-hard problem as it can also be reduced from the maximum coverage problem. Even the scheduling policy which always schedules maximum number of concurrent transmissions in each slot can not guarantee all transmission requests be satisfied. However, here we assume that the traffic load of the transmission requests are relatively low compared to the capacity of the network so that at least the scheduling policy that maximizes the concurrent transmissions can schedule all transmission requests in $T$ slots.

We leverage the heuristic solution of problem TJSPC to solve JSPC-TR. First consider the following problem:

$$\begin{aligned} \max\ gain = \quad & \textstyle\sum_{e\in E} min(\sum_k x_k S_k(e), \vec{b}(e)) \\ & -\beta \frac{\sum_k x_k C_k}{\sum_{e\in E} min(\sum_k x_k S_k(e), \vec{b}(e))} \\ \text{s.t.} \quad & \textstyle\sum_k x_k S_k(e) \geq b(e),\ \forall e \end{aligned} \tag{6.8}$$

In contrast to TJSPC, there are two differences. First since the transmission requests have to be satisfied, to minimize the energy cost, we need to choose more energy efficient sets. So we change the energy metric to energy efficiency metric which is the average cost of sending one packet. Second, there is no constraint on the total number of slots but the transmission request. This problem can be solved using the same greedy algorithm for TJSPC. Suppose for each $\beta$, the solution is $\vec{x}^{\beta}$. Define $E(\beta) = \sum_k x_k^{\beta} C_k$. Then we need to find an optimum $\beta$ that has the minimum energy cost:

$$\begin{aligned} & E' : \min_{\beta} E(\beta) \\ & \text{s.t. } \textstyle\sum_k x_k^{\beta} \leq T \end{aligned} \tag{6.9}$$

Suppose $\beta^*$ is the optimum $\beta$, then $\vec{x}^{\beta^*}$ is the heuristic solution to JSPC-TR. In next section, we discuss the algorithm to find the optimum $\beta^*$.

### 6.4.2 $\beta^*$-search Algorithm

Generally, as $\beta$ increases, equation 6.8 tends to find solutions that are more energy efficient thus $E(\beta)$ decreases. Theoretically, $E(\beta)$ is not a monotonically decreasing function. However, in all the simulations, we see a clearly decreasing trend. Therefore, heuristically, we will assume $E(\beta)$ is a decreasing function.

Consider a typical curve in Figure 6.3 [3] which shows the energy and number of slots used to transmit a transmission request. Let $T = 100$, so transmission request should be finished in 100 slots. As shown in the energy curve, the energy cost reduces as $\beta$ increases, however at the same time, the number of used slots also increase. The optimum operation point is point A in which exactly 100 slots are used and the energy cost is minimized. For practical purpose, we define a tolerance zone of width $\epsilon$, as shown in Figure 6.3. Here, $\epsilon$ is a protocol parameter that determine the converge rate of the protocol which we will show later. We denote $u$ as the number of used slots. The number of packets need to be transmitted in a transmission request $\vec{b}$ is $N = \sum_k \vec{b}(k)$.

From figure 6.3, we identify four characteristic operation regions(bounded by dotted line):

- small-$\beta$: $u < T - \epsilon$. In this sate, transmission requests are satisfied within T slots. The energy cost is high. It is clear that in order to reduce the energy cost, we need to increase $\beta$. However, this reduction must be performed carefully so that the transmission request is always satisfied. Intuitively, we need to achieve a balance between saving energy and satisfying transmission request. By invoking the fact that the relationship of $u$ vs. $\beta$, for $u < N$, is near linear, this prompts the use of the following increase strategy:

$$\beta_{i+1} = \frac{\beta_i}{2}(1 + \frac{u_i}{T})$$

  We will show later that such an update policy can reduce the energy cost while guaranteeing the transmission request satisfaction.

- opt-$\beta$: $T - \epsilon \leq u \leq T$. In this state, the network is operating within $\epsilon$ tolerance of the optimal point, where transmission request is satisfied and energy cost is a slightly higher. Hence the $\beta$ is left unchanged for the next frame:

$$\beta_{i+1} = \beta_i$$

[3] The figure is obtained by simulations discussed in section V.

- large-$\beta$: $T < u < N$. In this state, the network is operating in a region that not all transmission requests can be satisfied within T slots. It is clear that we need to decrease $\beta$ aggressively. Since the relationship of $u$ vs. $\beta$ is near linear, we use a decrease strategy as follows:

$$\beta_{i+1} = \beta_i \frac{T}{u_i} \delta_1$$

  We will show later by choosing $\delta_1 < 1$, we can guarantee that policy will converge to the opt-$\beta$ region.

- xlarge-$\beta$: $u \geq N$. In this sate, in all time slots, only one link is active. This consumes the least energy, however transmission request can not be satisfied within T slots when $T < N$. It is clear we need to decrease $\beta$ aggressively. However in this region, $u$ and $\beta$ is no longer linear and we have no idea how large $\beta$ is now. In order to converge to opt-$\beta$ region and guarantee transmission request, $\beta$ need to be decreased more aggressively than in the region of large-$\beta$:

$$\beta_{i+1} = \beta_i \frac{T}{u_i} \delta_2$$

  with $\delta_2 \leq \delta_1$.

We will show in next subsection that starting from any region, the above $\beta^*$-search algorithm converges to opt-$\beta$ region.

The entire JSPC-TR protocol is summarized in figure 6.4. The basic process is following: at a TDMA time frame, under the current $\beta$ and transmission request, the scheduler decides the state of the network then adjusts $\beta$ according to the $\beta^*$-search algorithm. The updated $\beta$ is then used for next TDMA frame. Here we assume the traffic requests change slowly compared to the converge rate of the $\beta^*$-search algorithm.

### 6.4.3 Analysis

First we present some analysis of the $\beta^*$-searching algorithm. Under the assumption of linear relationship of $u$ vs. $\beta$ in small-$\beta$ region/state, we are able to prove that network will converge to the opt-$\beta$ state. Another assumption is that the traffic load in TDMA frame does not change abruptly. The proof is similar to the one used in [69].

*Proposition 8:* Starting from small-$\beta$, with linear relationship between $u$ and $\beta$, the state will remain small-$\beta$ until it converges to opt-$\beta$ in $\lceil \frac{u_0 - 1}{\epsilon} \rceil$ iterations.

$N = \sum_e \vec{b}(e)$
Solve JSPC-TR using equation 6.8 with $\beta$
if $u < T - \epsilon$ /*State is small-$\beta$*/
$\beta = \frac{\beta}{2}(1 + \frac{u}{T})$
else if $T - \epsilon \leq u \leq T$ /*State is opt-$\beta$*/
$\beta = \beta$
else if $T < u < N$ /*State is large-$\beta$*/
$\beta = \beta \frac{T}{u} \delta_1$
else if $T \geq N$ /*State is xlarge-$\beta$*/
$\beta = \beta \frac{T}{u} \delta_2$
end

Figure 6.4: JSPC-TR protocol and $\beta^*$-search algorithm.

*Proof:* Suppose the linear behavior for $u < T - \epsilon$ is $u = a\beta$ and $u_i < T - \epsilon$. So the $\beta$ is increased by:

$$\beta_{i+1} = \frac{\beta_i}{2}(1 + \frac{T}{u_i})$$

Thus,

$$u_{i+1} = \frac{u_i}{2}(1 + \frac{T}{u_i}) = \frac{u_i + T}{2}$$

Since $\beta_{i+1} > \beta_i$, the next state can either be small-$\beta$, opt-$\beta$, large-$\beta$ or xlarge-$\beta$. Suppose the next state is neither small-$\beta$ nor opt-$beta$, then $u_{i+1} > T$. Then,

$$u_{i+1} = \frac{u_i + T}{2} > T$$

Hence, $u_i > T$. However this contradicts with $u_i < T - \epsilon$ since the starting state is small-$\beta$. Thus, the state can only be small-$\beta$ before it reaches opt-$\beta$.

Now we prove that the converge takes $\lceil \frac{u_0 - 1}{\epsilon} \rceil$ iterations. Let $j$ be the first one when the network is in opt-$\beta$ state.

$$\begin{aligned} u_j &= \frac{u_{j-1} + T}{2} > T - \epsilon \\ u_{j-1} &= \frac{u_{j-2} + T}{2} > T - 2\epsilon \\ &\vdots \\ u_1 &= \frac{u_0 + T}{2} > T - 2^{j-1}\epsilon \end{aligned}$$

Thus, it takes $j > log_2(\frac{u_0 - 1}{2})$ iterates before $u_j > T - \epsilon$. In the whole process, the transmission request is always guaranteed.

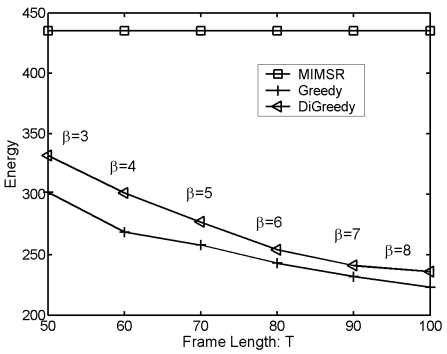



Figure 6.5: Energy cost reduces as latency constraint increases as a function of $T$ with varying $\beta$.

■

*Proposition 9:* Starting from large-$\beta$ or xlarge-$\beta$, the state will converge to opt-$\beta$.

*Proof:* Suppose the linear behavior for $u < T - \epsilon$ is $u = a\beta$. For large-$beta$ state, $u > T$. So $\beta$ is decreased by:

$$\beta_{i+1} = \beta_i \frac{T}{u_i}\delta_1 = \beta_0 \frac{T^i}{\prod_{k=0}^{i} u_k}\delta_1^i$$

Since $\delta_1 < 1$ and $u_k > T$, $\beta$ will keep decreasing until it change to either opt-$\beta$ region or small-$\beta$ region which will converge to opt-$\beta$ by Lemma 8.

Similarly, starting from xlarge-$\beta$, the network can also converge to opt-$\beta$. ■

## 6.5 Simulation Results

We simulate the performance of the algorithms for a stationary network consisting of a grid of 49 nodes. The distance between adjacent nodes is set to 20 meter. The radio parameters are set according to the CC1000 radio used in Mote MICA2 [70, 3]. The minimum transmission power is $P_{min} = -20dBm$ and the maximum transmission power is $P_{max} = 5dBm$. According to [68], The path loss factor in a typical outdoor environment is 4 and the noise floor is around $-105dBm$. The $SNR$ threshold $\gamma$ for successfully packet reception is set to be $10dB$. We choose 42 links and pre-computed all feasible transmission sets and their energy costs. The maximum number of active links in a transmission scenario is 5.

### 6.5.1 Simulation Results for TJSPC

Besides the Greedy and DiGreedy algorithms, we also simulate the scheduling algorithm (referred as MIMSR) proposed in [17]. We simulate 20 time frames which consist of $T$ slots. Each node randomly

generates 1 to 6 packets to be transmitted in each time frame. All results are averaged over 10 seeds. From the simulations, we learned three key lessons, described below.

*Lesson 1: By relaxing the latency bound, we can get significant energy savings*[4]. In the simulation, we fix the traffic load while increasing the latency bound $T$. Figure 6.5 shows the impact of $T$ on the energy cost performance of the algorithms. The packet reception ratio (which is directly proportional to the throughput) remains above $95\%$ for all $T$ and $\beta$. Larger $\beta$ can be used for higher latency bound because preference can be given to feasible transmission set with smaller energy cost. As $T$ increases, the energy cost of Greedy and DiGreedy decreases significantly. Compared to MIMSR which remains around 435 regardless of $\beta$, the savings can be as high as 50%.

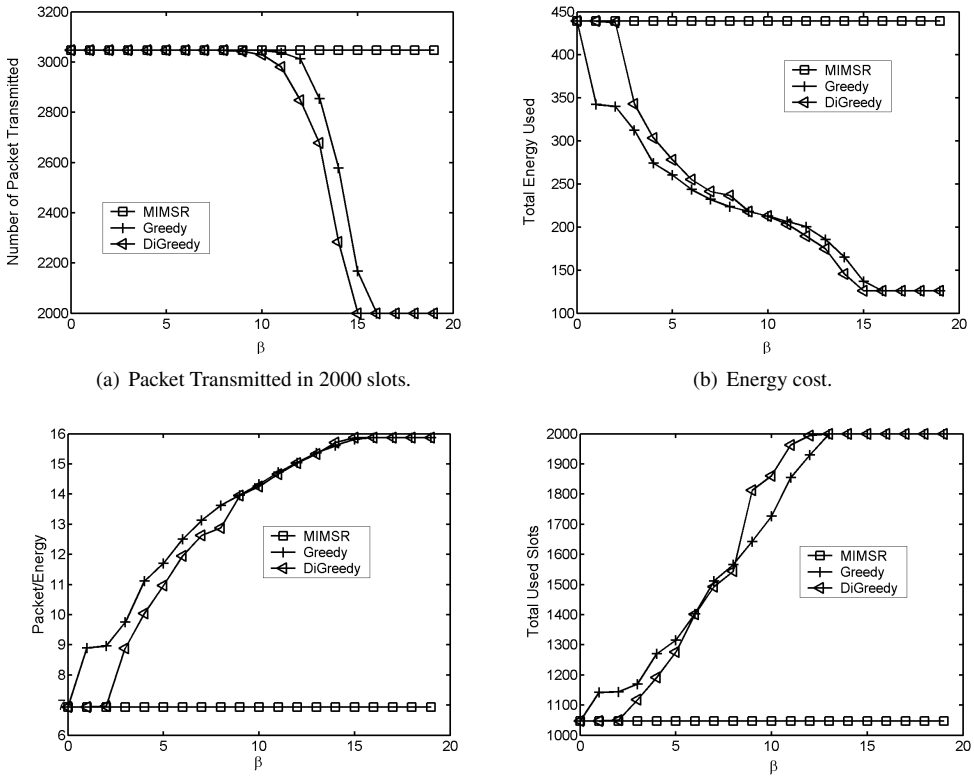
Figure 6.6(d) shows that total number of used slots for the algorithms with the same traffic load and fixed $T = 100$. As MIMSR always schedules the maximal feasible set, it uses less slots in transmitting the traffic. However, by increasing $\beta$, Greedy and DiGreedy would give higher and higher preference on low energy cost transmission sets, thus increase the number of slots used. The more slots used means more packets will be transmitted at the end of a time frame, thus a higher average latency, but still within the latency bound.

Go back to figure 6.2(a) and 6.2(b) which show the schedules computed by $\beta = 0$ and $\beta = 10$ separately. Clearly $\beta = 10$ is able to choose more energy efficient transmission sets.

*Lesson 2: By varying $\beta$, the algorithm is able to save significant energy without hurting throughput.* Figure 6.6(a) and 6.6(b) shows the number of packets delivered and the total energy cost in 20 frames which consists of 100 slots respectively. When $\beta \leq 10$, Greedy and DiGreedy can deliver almost same number of packets. The energy cost decreases as $\beta$ increases even though the number of packets delivered is the same. The energy savings can be as much as $50\%$. This shows the algorithms' ability to choose a better combination of transmission scenarios. When $\beta > 10$, the number of packets delivered by Greedy and DiGreedy begins to decrease. When $\beta > 15$, the algorithms will always choose the transmission scenario with only one link active that is most energy efficient. Thus the total number of packets can be delivered in 2000 slots remains 2000. Figure 6.6(c) shows the energy efficiency in terms of the number of packets delivered in units of energy. Clearly as $\beta$ increases, the energy efficiency of the scheduled set increases.

*Lesson 3: DiGreedy algorithm has comparable performance to the Greedy approximation algorithm.* For all the simulations, Greedy and DiGreedy can save more energy than MIMSR while maintaining relatively same throughput or at a little sacrifice of the throughput. As we can see from all

[4]The unit of energy is $12.7\tau$ mJ, where $\tau$ is the transmission time of one packet




(a) Packet Transmitted in 2000 slots.

(b) Energy cost.

(c) Energy efficiency in terms of number of packets transmitted per unit of energy.

(d) Total used slots for delivered packet.

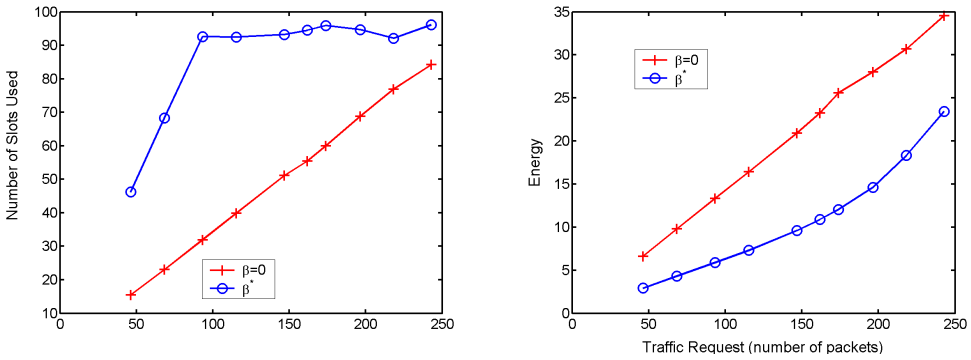
Figure 6.6: Performance of MIMSR, Greedy and DiGreedy as a function of $\beta$ with $T = 100$.

figures, DiGreedy, as a heuristic solution with no approximation guarantee, has almost the same performance as the Greedy which is $(1 - \frac{1}{e}$ approximate to the optimization solution.

### 6.5.2 Simulation Results for JSPC-TR

We simulated the $\beta^*$-searching algorithm under various traffic load requests to find the $\beta^*$. Then we compared the performance of two different schedules computed by $\beta = 0$ and $\beta^*$. All results are averaged over 10 seeds. In the simulation, $\delta_1 = \delta_2 = 0.8$ and $\epsilon = 10$.

Figure 6.7(a) and 6.7(b) show the number of slots and energy used to finish the transmission request under different traffic load separately. Under all traffic load request, our algorithm is able to operate in opt-$\beta$ region and thus consume much less energy while all transmission requests are satisfied within $T = 100$ slots. The number of slots used by $\beta^*$ are always between 90 and 100, except for very low traffic load when the number of packets is less than 100.




(a) Number of slots used (b) Energy cost

Figure 6.7: Performance under various traffic request load.

## 6.6 Discussion

In EEJSPC, we studied the fundamental energy efficiency problem of joint TDMA link scheduling and power control in wireless sensor networks. We found that different transmission scenario can have significantly different total transmission powers. By carefully choosing different combinations of feasible transmission scenarios in multiple slots, the total energy costs can be reduced. This improves energy efficiency compared to previously proposed joint scheduling and power control algorithms, which always try to schedule maximum concurrent transmissions.

We formulated a joint link scheduling and power control problem that aims to maximize a function of throughput and energy cost subject to latency constraint(TJPSPC). This formulation allows a tunable performance tradeoffs between throughput, latency and total energy cost. We showed this NP-hard problem formulation can be solved using a greedy algorithm which is an $(1 - \frac{1}{e})$-approximation to the optimal solution with exponential approximation. We then presented DiGreedy, a heuristic greedy algorithm with polynomial complexity. Simulation results show that DiGreedy algorithm has similar performance to the greedy algorithm, and can achieve significant energy savings at no or little sacrifice of the throughput. We also investigated the joint scheduling and power control problem with constraint on the number of packets to be sent on each link. We leverage the heuristics for TJSPC to solve this problem by using the optimum $\beta$ which achieves energy efficiency while guaranteeing the satisfaction of transmission requests. Simulation results show 50% energy savings can be achieved without sacrificing throughput.

## 6.7 Appendix

*Proposition 10:* $wt(\vec{x}) \geq [1-(1-\frac{1}{k}^k]wt(OPT) > (1-\frac{1}{e})wt(OPT)$

*Proof:* Suppose $\vec{x}^i$ is the greedy solution in the first $i$ slots, let

$$G_i = \sum_k x_k^i S_k$$

and

$$wt(G_i) = \sum_e min(\sum_k x_k^i S_k, \vec{b}(e)) - \beta \sum_k x_k^i c_k$$

Suppose in $i+1$ slot, transmission set $S_j$ is chosen, Then $x_j^{i+1} = x_j^i + 1$ and $G_{i+1} = G_i \cup S_j$, we have:

$$wt(G_{i+1}) = \sum_e min(\sum_k x_k^{i+1} S_k, \vec{b}(e)) - \beta \sum_k x_k^{i+1} c_k$$

Now, the gain of the first selected $(i-1)$ sets is $wt(G_{i-1})$. The difference between $wt(G_{i-1})$ to the gain of the optimal solution is $wt(OPT) - wt(G_{i-1})$. Then at least $wt(OPT) - wt(G_{i-1})$ worth of gain not covered by the first $(i-1)$ sets are covered by the $T$ sets of $OPT$. By the pigeonhole principle, one of the $T$ sets in the optimal solution must cover at least $\frac{wt(OPT)-wt(G_{i-1})}{T}$ worth of gain. Since $S_j$ is a set that achieves maximum additional gain, it must also cover at least $\frac{wt(OPT)-wt(G_{i-1})}{T}$. That is:

$$wt(G_i) - wt(G_{i-1}) \geq \frac{wt(OPT) - wt(G_{i-1})}{T}$$

Now let us suppose for $i = 1$, $wt(G_1) \geq \frac{wt(opt)}{T}$, then,

$$\begin{aligned}
wt(G_{i+1}) &= wt(G_i) + (wt(G_{i+1}) - wt(G_i)) \\
&\geq wt(G_i) + \frac{wt(OPT) - wt(G_i)}{T} \\
&= (1 - \frac{1}{T})wt(G_i) + \frac{wt(OPT)}{T} \\
&\geq (1 - \frac{1}{T})(1 - (1 - \frac{1}{T})^i)wt(OPT) + \frac{wt(OPT)}{T} \\
&= (1 - (1 - \frac{1}{T})^{i+1})wt(OPT) \\
&> (1 - \frac{1}{e})wt(OPT)
\end{aligned}$$

■

# Chapter 7

# Conclusions

## 7.1 Summary

In this thesis, we discussed the energy latency tradeoffs for medium access and sleep scheduling in wireless sensor networks and presented MAC protocol and scheduling algorithms for four specific application scenarios, which could achieve the balance between energy and latency under the different application requirements. We summarize our contributions in this chapter.

Previous sensor network MAC protocols save energy by sacrificing the latency performance, which motivated us to design energy efficient yet also low latency MAC for wireless sensor networks. With different application scenarios, we have studied four problems and shown that energy efficient MAC protocols can be designed for wireless sensor networks without necessarily sacrificing application-specific latency performance.

For contention-based MAC, we first show that previously proposed MAC protocols for sensor networks that utilize activation/sleep duty cycles suffer from a *data forwarding interruption problem*. In these schemes, not all nodes on a multihop path to the sink can be notified of data delivery in progress, therefore intermediate nodes may go to sleep and can not help forward the packets, resulting in significant sleep delay. By giving the active/sleep schedule of a node an offset that depends upon its depth on the tree, DMAC allows continuous packet forwarding because all nodes on the multihop path can be notified of the data delivery in progress in a pipeline way. DMAC also adjusts node duty cycles adaptively according to the traffic load in the network by varying the number of active slots in an schedule interval. We further propose a *data prediction* mechanism and the use of *more to send* (MTS) packets in order to alleviate problems pertaining to channel contention and collisions. Our simulation results show that by exploiting the application-specific structure of data gathering trees in sensor networks, DMAC provides significant energy savings and latency reduction while ensuring high data reliability.

The second study, DESS, aims to minimize the worst communication latency given that each sensor has a duty cycling requirement of being awake for only $\frac{1}{k}$ time slots on an average. As a first step we consider the single wake-up schedule case, where each sensor can choose exactly one of the $k$ slots to wake up. We formulate a novel graph-theoretical abstraction of this problem in the general setting of a low-traffic wireless sensor network with arbitrary communication flows and prove that minimizing the end-to-end communication delays is in general NP-hard. However, we are able to derive and analyze optimal solutions for two special cases: tree topologies and ring topologies. Several heuristics for arbitrary topologies are proposed and evaluated by simulations. Our simulations suggest that distributed heuristics may perform poorly because of the global nature of the constraints involved.

The third study, MLSR, considers the problem of minimizing the average communication latency for only the active flows to the base station in the network. Since the typical flows in wireless sensor network are predictable and long-lived, it is possible to design routing paths of the flows and the on/off schedules of the nodes on the paths to minimize the average latency for all the active flows. Clearly the scheduling and routing are closely coupled together, thus we formulated a joint scheduling and routing problem with objective to find the minimum latency joint schedule and route for current active flows. By constructing a novel delay graph, the problem can be solved optimally by $M$ node-disjoint paths algorithm under FDMA channel model. We further extended the algorithm to handle dynamic traffic changes and topology changes in wireless sensor networks. We also proposed a heuristic solution for the minimum latency joint scheduling and routing problem under single channel interference.

In fourth study, we investigate the fundamental issue of TDMA link scheduling with transmission power control using a realistic SINR-based interference model. We formulate it as a novel optimization problem (TJSPC) that provides tunable tradeoffs between energy , throughput, and latency, through a single parameter $\beta$. We present a centralized greedy algorithm for this problem that has a provable $(1-\frac{1}{e})$-approximation guarantee, along with a good distributed heuristic. We evaluate the performance of these algorithms through simulations. Our results show that for moderate traffic loads, with appropriate tuning of parameter $\beta$, major energy savings can be obtained without significantly sacrificing throughput. We further proposed the minimum energy joint scheduling and power control problem (EEJSPC) under throughput and latency constraints. We designed distributed iterative approach which leverages the heurisitcs for TJSPC and converges to the optimal $\beta$ in $O(\frac{1}{epsilon})$ steps.

All four case studies show that under specific application scenarios, it is possible to design both energy efficient and low latency medium access and sleep scheduling suitable for the specific sensor network application.

## 7.2 Future Directions

There are several research directions for the work presented in this thesis. Here we briefly discuss three possible extensions: the integration of information processing in general sleep scheduling techniques, different latency metrics in DESS and MLSR, the adaptive transmission rate and end-to-end latency problem in EEJSPC.

1. A specific feature of wireless sensor network is its data-centric paradigm [32, 92, 95, 96], as opposite to the address-centric or node-centric paradigm. Physical samples collected by sensors nearby are often strongly correlated. In-network processing, such as compression and signal processing is useful to eliminate redundant data, thus saves the information to be sent to the sink. A data centric routing scheme could affect the efficiency of data aggregation achieved in the network. For example, the authors in [85] analyzed the performance of routing with compression in wireless sensor networks. A sleep scheduling scheme changes the network topology from time to time, leading to re-construction of the routing paths. Yet it is not clear how to integrate the data centric routing schemes with sleep scheduling algorithms to choose the right node to sleep, the right time to sleep and the right path for data compression to satisfy the energy and latency requirements of the application.

2. In DESS, the objective function is to minimize the worst case latency while in MLSR the objective is to minimize the average latency. There are other latency performance metrics. One example is that the flows in the network have different latency deadline requirements when the physical events the network detected require different respond speeds. Even for flows with same latency deadline, the flow with source closer to the sink can tolerate higher sleep latency per hop. Thus a different sleep scheduling problem would be to find schedules for nodes in the network to satisfy each flow's latency deadline with minimum energy.

3. In the work of EEJSPC, we assumed a fixed data rate and SINR threshold of wireless radio. Many modern wireless radios [3], however, are capable of supporting multiple data rates. There are many research works [91, 94, 97, 101, 102] on energy efficient scheduling of dynamic modulation. Normally low data rate modulation is less efficient in utilizing channel bandwidth but more robust to noise and interference, thus can work under low SINR environment. High data rate modulation, however, is more bandwidth efficient yet less robust to channel error, thus requires high SINR and high transmission power. We need to extend the model in EEJSPC to take the multiple data rates and SINR thresholds into consideration.

Another possible extension of EEJSPC is to consider the end-to-end delay of flows. In EEJSPC, we assumed a per hop latency bound which can be calculated based on end-to-end latency and the routing paths. A better scheme is to directly consider the end-to-end latency of the traffic flows. Packets can have different per hop latency bounds. Also the same packet can have different hop latency bounds at different hops. This, along with the multiple data rates and SINR thresholds, creates a larger design space to explore better energy latency tradeoffs.

# Appendix A

# Wakeup Radio

## A.1 Overview

Wireless sensor networks have limited energy resource and need to save energy as much as possible. As idle listening in radios has been identified as a major source of energy wastage, energy efficient communication protocols turn off the radio to save energy when a node is not in sending/receiving states or does not detect any event. However, this creates problems with communication. Often it is necessary for nodes that are not directly involved with sensing an event to be involved in communication later, such as a multihop relay node. Thus some wakeup schemes are needed in the network to wake up a sleeping node when the node is needed for communication.

There are two types of wakeup protocols: active wakeup protocol and on demand wakeup protocol. In active wakeup protocols, nodes follow a pre-determined periodical wakeup/sleep schedule so that when nodes enter sleep mode, they schedule a timer to wake up at a pre-determined time, which can be synchronous [60] or asynchronous [62]. In on demand wakeup protocol, a sleeping node can be woken at any time via an out-of-band channel, such as wake-on-wireless [51], STEM [54] and PicaRadio [75]. There are also hybrid schemes such as those presented in [63] and [53].

Both active and on demand wakeup schemes have great potential in energy saving for sensor networks when communications happen infrequently. However, the wakeup schemes have the disadvantage of increased latency. In active protocols, the periodical wakeup/sleep schedule in active wakeup protocols cause sleep latency that is proportional to the number of hops with a slope of the duration of the pre-determined schedule iteration. Our works on DMAC [46] and DESS [48] identified this problem and reduced the sleep latency to achieve low latency while keeping the same energy efficiency of the periodical wakeup/sleep schedule. The on-demand approaches, at the cost of additional hardware,

Table A.1: The measured average power consumption of the MiniBrick

| Mode | Sleep | Receive | Transmit |
|---|---|---|---|
| Current(mA) | 0.6 | 2.2 | 2.4 |
| Power(mW) | 2.0 | 7 | 8 |

potentially could have smaller latency. However, due to current radio technology limitations, either non-negligible energy cost or latency still exists in the on-demand wakeup protocols. In this chapter, we will discuss the energy-latency tradeoffs for on-demand wakeup protocols. *We argue that our low-latency and energy-efficient sleep scheduling algorithms can be employed on top of the on-demand protocols to achieve better energy-latency tradeoffs.*

On-demand wakeup schemes need an additional wakeup radio besides the data radio. Figure A.1 shows the communication abilities and energy cost of the radios. Current data radios have better communication abilities such as high data rate and long distance, but also with high idle listening power consumption. The wakeup radios have limited communication abilities, such as detecting busy tone [53] or frequency identification [58]. The wakeup radio uses much less power compared to the data radio via either a low duty cycle [53, 54] or hardware design [56].

## A.2 Preliminary Wakeup Radio

The simplest way to implement a wakeup radio is to have the wakeup radio active all the time [51]. When a node need to communicate with a neighbor node, it sends a wakeup signal which may be a short impulse or a short message. However, due to current radio abilities, the idle listening of the wakeup radio still has non-negligible energy cost. For applications that need to operate for months or even years, the energy cost from the wakeup radio can not ignored. The authors in [51] built a prototype wakeup radio which is called MiniBrick. Table A.1 shows the power consumption of MinBrick. Compared to the power consumption of Lucent Orinoco card shown in table A.2, the power consumption is very low. However, compared to the radio in MICA2 mote shown in table A.3, the power consumption of MiniBrick can not be ignored. If MiniBrick is used as a wakeup radio in Mica2, it can not be active all the time otherwise energy will be drained out quickly. As long as the power consumption of the wakeup radio is not negligible, it is still necessary to preserve energy of the wakeup radio. The best way again is to turn the wakeup radio off. Thus active scheduling algorithms can still be employed in this category of wakeup radio.

Table A.2: The average power consumption of the Lucent Orinoco WLAN card

| Mode | Sleep | Receive | Transmit |
|---|---|---|---|
| Current(mA) | 10 | 180 | 280 |
| Power(mW) | 50 | 900 | 1400 |

Table A.3: The average power consumption of CC1000 in Mica2

| Mode | Sleep | Receive | Transmit |
|---|---|---|---|
| Current(mA) | 0.6 | 7.4 | 5.6 |
| Power(mW) | 1.8 | 22.2 | 16.8 |

## A.3 Periodical Active/Sleep Wakeup Radio

Some works [52, 53, 54] assume the wakeup radio achieves ultra low power by turning off wakeup radio periodically according to a pre-defined sleep schedule, same as the sleep schedule on the data radio. When a node has a message to its destination, it sends a wakeup signal on the wakeup radio that is long enough for the receiver's wakeup radio to detect the wakeup signal during its active period, shown in Figure A.2. Some other works [64, 65, 66] proposed Preamble Sampling or Low Power Listening scheme, in which the receiver periodically wake up to sample the channel. If a node wish to transmit, it sends a preamble that is long enough so that the receiver can detect the preamble. Then the receiver will be full active to receive the packet following the preamble. The wakeup signal can be sent over a high level interface and directly in the physical layer which is more energy efficient.

Depending on the communication ability of the wakeup radio, there are broadcast wakeup or directed wakeup signals. If the destination ID can be encoded into the wakeup signal and the wakeup radio is able to decode it, then only the intended receiver need to be waken up. If the wakeup signal can only detect a busy tone by the energy threshold, then the entire neighborhood the sender will be waken up.

In broadcast wakeup schemes, because the wakeup radio only needs to detect a wakeup signal (e.g. busy tone) by an energy threshold, it does not need sophisticated circuit to decode a message thus the detection time of the wakeup radio can be designed to be very short (e.g. 1ms). Thus a short schedule iteration can be employed to reduce latency. However a broadcast wakeup signal will wake up entire neighborhood, in which many nodes does not need to be waken up. Although a filter packet can be sent later to put non-receiver nodes back to sleep, the switch on/off overhead could be high (e.g. in CC1000, the delay is about 4ms and power consumption is about 20mW). In a dense sensor network, this could incur significant energy wastage when only a small part of the nodes need to be active. In directed wakeup schemes, as the wakeup radio is designed to wake up only the intended receiver, the active time

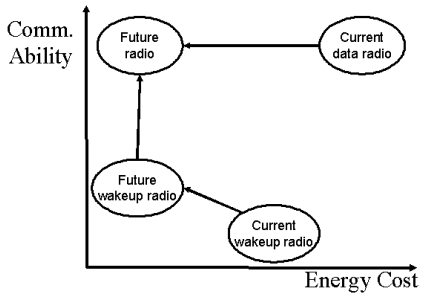



Figure A.1: Energy and communication abilities of radios

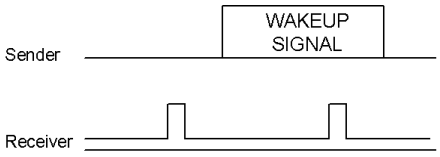



Figure A.2: Wake up process in the periodical active/sleep wakeup channel

of the wakeup radio has to be significantly long to receive a message, so is the total schedule iteration. Thus again because of the periodical sleep schedule of the wakeup radio, there will be sleep latency on the wakeup channel.

STEM [54, 55] investigated both the directed wakeup scheme STEM-B and broadcast wakeup scheme STEM-T. While STEM-B is more energy efficient than STEM-T, it has significant longer latency on the wakeup channel than STEM-B. Authors in [52] proposed a scheme to schedule the wakeup radio ahead of data radio in a similar way to DMAC to reduce the wakeup latency by pipeline the wakeup signal. [52] used a broadcast wakeup scheme but sent a filter packet after entire neighbor nodes are waken up. Then during the data transmission from the sender to the receiver on the data channel, the receiver will send busy tone to wake up its own neighborhood on the wakeup channel. Then when the receiver received the packet, it can immediately send a filter packet to its neighborhood and data packet to its intended next hop without sleep latency. Thus, we believe the sleep schedule algorithm we studied on data radio can also be employed on the wakeup channel to achieve better energy latency tradeoffs.

## A.4 Ultra Low Power Wakeup Radio

There are also works on design radio hardware with ultra low power consumption that the wakeup radio can be kept active all the time, such as PicoRadio [56, 75] and RTID [57]. RTID designed a radio-triggered hardware that is able to extracting energy from the radio signal to provide wake-up signals to the network node. However their wakeup radio acts as a mechanism to wakeup the CPU of the node,

Table A.4: The wakeup radio technology

| Wakeup Radios | Preliminary | Intermediate | Premium | Future |
|---|---|---|---|---|
| Power Consumption | Non-negligible | Non-negligible | Ultra low | Ultra low |
| Active Schedule | Always on | Periodical on/off | Always on | Always on |
| Addressibility | Directed | Broadcast or Directed | Broadcast | Directed |
| Strongpoint | No sleep latency | Low energy consumption of wakeup radio | No sleep latency and low energy cost | No sleep latency and low energy cost |
| Weakness | High energy cost | Latency caused by periodical schedule | Overhead of waking up all neighbors | Contention in data radio |
| Role of DESS | DESS can schedule wakeup radio sleep without hurting latency | DESS can reduce sleep latency on wakeup radio | The schedule of DESS can avoid waking up entire neighborhoods | DESS can be incorporated into TDMA to reduce end-to-end sleep latency |

not an intended receiver of a message. And the event on RTID can only be some radio messages instead of environment stimulant. The PicoRadio is not addressible, so it can be only designed as a broadcast mechanism which is not suitable for a dense wireless sensor network. Currently, the PicoRadio is still a prototype and has not been fully implemented.

Even if we assume an ultra low power wakeup radio that can be kept active all the time and is addressible exists, there is a potential problem of contention under medium or high traffic load. First there are contentions on the wakeup channel. Because the wakeup signals contain the ID of the intended receivers, they must be correctly received by the receivers without collision. If in a single broadcast domain, multiple senders need to wakeup their receivers, the contentions on the wakeup channel could cause significant collisions of the wakeup signals because of the limited MAC ability of the wakeup radio. Either unrelated nodes have to be waken up or the intended receiver will miss the packet depending on the handling of a corrupted wakeup signal. For example, STEM-B [54, 55] is a directed wakeup scheme. When there is collision in the wakeup signals, a node detected a collision packet will wake up for a sufficient period and go back to sleep if it is not the intended receiver.

We further assume that the wakeup radio can employ CDMA schemes, so there is no contention in the wakeup channel. However, there are contention on the high rate data radio. Suppose multiple intended receivers are waken up and their senders will transmit the data messages to them. Under synchronized traffic load [61] which is caused by the same event, all the nodes are waken up at the same time and thus will contend for the data channel which could cause significant collision and energy

cost. The TDMA-like contention free MAC can potentially eliminate the contention. However, since all nodes are assigned active slots to communicate, a node would have to wait for its intended receiver's active slot to transmit the data, which will cause the sleep latency. Previous TMAC protocols only focused on minimizing the length of a time frame in which each node is assigned at least one active slot. This can minimize the hop-by-hop latency, however, does not consider the increased end-to-end latency caused by sleep scheduling. The work of DESS can be combined into the design of TDMA slot assignment to reduce the sleep latency, while maintaining contention free. One of our future work is to incorporate some mechanisms into the sleep scheduling algorithm to reduce the possible contentions of the synchronized traffic load.

As the radio technology evolves, when the radio is able to send/receive long frame at high data rate with negligible idle listening power, such as the future radio indicated in Figure A.1, then there is no need to turn off radio to save communication. The radio can be used for data transmission and be kept active all the time.

# References


[1] D. Estrin , J. Heidemann , R. Govindan and S. Kumar, "Next Century Challenges: Scalable Coordination in Sensor Networks" , *ACM MobiCom*, August 1999

[2] ATMEL Atmega128 Datasheet, http://www.atmel.com

[3] Chipcon CC1000 Datasheet, http://www.chipcon.com

[4] L. M. Feeney, M. Nilsson, "Investigating the Energy Consumption of Wireless Network Interface in an Ad Hoc Networking Environment", *IEEE INFOCOM* 2001

[5] M. Stemm and R. H. Katz, "Measuring and reducing energy consumption of network interfaces in hand-held devices,", *IEICE Transactions on Communications*, vol. E80-B, no.8, pp.1125-1131, Aug. 1997

[6] L. Pond and V.O.K. Li, "A Distributed Time-Slot Assignment Protocol for Mobile Multi-Hop Broadcast Packet Radio Networks", *Proceedings of IEEE Military Communications Conference (MILCOM)*, October 1989.

[7] R. Ramaswami and K. K. Parhi, "Distributed Scheduling of Broadcasts in a Radio Network", *IEEE INFOCOM*, April 1989.

[8] I. Cidon and M. Sidi, "Distributed Assignment Algorithms for Multi-Hop Packet-Radio Networks", *IEEE Transactions on Computers*, vol. 38, no. 10, pp. 1353-1361, October 1989.

[9] K. Sohrabi, J. Gao, V. Ailawadhi, and G.J. Pottie, "Protocols for Self-Organization of a Wireless Sensor Network", *IEEE Personal Communications*, Vol. 7, No. 5, Oct. 2000

[10] G. Pei and C. Chien, "Low power TDMA in Large Wireless Sensor Networks", *IEEE MILCOM* Vol. 1, 28-31 Oct. 2001

[11] C. Schurgers, V. Tsiatsis, M. B. Srivastava, "STEM: Topology Management for Energy Efficient Sensor Networks", *IEEE Aerospace Conference*, 2002

[12] S. Coleri, A. Puri, and P. Varaiya, "Power Efficient system for Sensor Networks", *Proceedings of Eighth IEEE International Symposium on Computers and Communication (ISCC)*, July 2003.

[13] V. Rajendran, K. Obraczka, and J.J. Garcia-Luna-Aceves, "Energy-Efficient, Collision-Free Medium Access Control for Wireless Sensor Networks", *Proceedings of ACM Sensys*, November 2003.

[14] J. Zander, "Distributed cochannel interference control in celluar radio systems", *IEEE Trans. Veh. Technol.* vol.41 pp. 305-311 Aug. 1992.

[15] T. ElBatt, A. Ephremides, "Joint Scheduling and Power Control for Wireless Ad-hoc Networks", *IEEE Infocom*, 2002

[16] T. ElBatt, A. Ephremides, "Joint Scheduling and Power Control for Wireless Ad-hoc Networks", *IEEE Transaction on wireless communications*, vol 3. No.1 pp. 74-85, Jan. 2002

[17] K. Wang, C. F. Chiasserini, R.Rao and J. G. Proakis, “A Distributed Joint Scheduling and Power Control Algorithm for Multicasting in Wireless Ad Hoc Networks”, *IEEE ICC*, 2003

[18] R. Bhatia, M. Kodialam, “On Power Efficient Communication over Multi-hop Wireless Networks: Joint Routing, Scheduling and Power Control”, *IEEE Infocom*, 2004.

[19] R.L. Cruz, A. V. Santhanam, “Optimal Routing, Link Scheduling and Power Control in Multi-hop Wireless Networks”, *IEEE Infocom*, 2003.

[20] A. Behzad, I. Rubin and A. Mojibi-Yazdi, ”Distributed Power Controlled Medium Access Control for Ad-Hoc Wireless Networks”, *Proceedings of the 18th IEEE Annual Workshop on Computer Communications (CCW)* October, 2003.

[21] W. R. Heinzelman, A. Chandrakasan and H. Balakrishnan, “Energy Efficient Communication Protocol for Wireless Microsensor Networks”, *Proceedings of IEEE HICSS*, January 2000.

[22] S. Bandyopadhyay and E. J. Coyle, “An Energy Efficient Hierarchical Clustering Algorithm for Wireless Sensor Networks”, *Proceddings of IEEE Infocom*, April, 2003.

[23] D. S. Hochbaum, “Approximation Algorithms for NP-hard Problems”, *PWS Publishing Company*, pp. 135-138

[24] W. Ye, J. Heidemann, and D. Estrin, “Medium Access Control with Coordinated, Adaptive Sleeping for Wireless Sensor Networks”, Technical Report USC ISI-TR-567, January, 2003. (Accepted to appear in *ACM/IEEE Transactions on Networking*.)

[25] J. Li, C. Blake, D. Couto, H. Lee and R. Morris, “Capacity of Ad Hoc Wireless Networks”, *ACM Mobicom* July 2001

[26] T. V. Dam, K. Langendoen, “An Adaptive Energy-Efficient MAC Protocol for Wireless Sensor Networks”, *ACM Sensys* Nov. 2003

[27] J. Elson, L. Girod and D. Estrin, “Find-Grained Network Time Synchronization using Reference Broadcasts”, *ACM SIGOPS* 2002

[28] E. Jung, N. H. Vaidya, “An Energy Efficient MAC Protocol for Wireless LANs”, *IEEE Infocom* 2002

[29] V. Raghunathan, C. Schurgers, S. Park, and M. B. Srivastava, “Energy-aware wireless microsensor networks”, *IEEE Signal Processing Magazine* 2002

[30] Y. Li, W. Ye, J. Heidemann “Schedule and Latency Control in S-MAC”, Poster, *UCLA CENS research review* 2003

[31] Y. Yu, B. Krishnamachari, and V. K. Prasanna, ”Energy-Latency Tradeoffs for Data Gathering in Wireless Sensor Networks”, *IEEE InfoCom*, Mar. 2004

[32] B. Krishnamachari, D. Estrin and S. Wicker, “The impact of data aggregation in wireless sensor networks”, *International Workshop on Distributed Event-based Systems*, 2002

[33] C. S. Raghavendra and S. Singh, “PAMAS-power aware multi-access protocol with signaling for ad hoc networks”, *Computer Communication Revies*, 1998

[34] V. Bharghavan, A. Demers, S. Shenker, and L. Zhang, ”MACAW: A Media Access Protocol for Wireless LAN’s,” *ACM SIGCOMM*, 1994

[35] MOTE, http://www.xbow.com

[36] O. Dousse, P. Mannersalo, P. Thiran, “Latency of Wireless Sensor Networks with Uncoordinated Power Saving Mechanisms”, *MOBIHOC* 2004.

[37] M. R. Garey and D. S. Johnson, “Computers and Instractability: A Guide to the Theory of NP-Completeness”, W.H Freeman, San Francisco, CA, 1979.

[38] W. Liang, “Constructing Minimum Energy Broadcast Trees in Wireless Ad Hoc Networks”, *MOBIHOC* 2002.

[39] J. Gruenen and J. Rabaey, “Lightweight time synchronization for sensor networks”, *ACM WSNA*, 2003.

[40] S. Ganeriwal, R. Kumar, and M.B. Srivastava, “Timing-Sync Protocol for Sensor Networks”, *ACM SenSys*, 2003.

[41] D. Ganesan, B. Krishnamachari, A. Woo, D. Culler, D. Estrin and S. Wicker, “Complex Behavior at Scale: An Experimental Study of Low-Power Wireless Sensor Networks”, UCLA CS Technical Report UCLA/CSD-TR 02-0013, 2002.

[42] J. Zhao and R. Govindan, “Understanding Packet Delivery Performance in Dense Wireless Sensor Networks”, *ACM Sensys*, November 2003.

[43] A. Woo, T. Tong, and D. Culler, “Taming the Underlying Issues for Reliable Multhop Routing in Sensor Networks”, *ACM SenSys*, November 2003.

[44] G. Lu, C. S. Raghavendra, “Improving the performance of IEEE 802.11 in Wireless Multihop Networks”, *IEEE MWCN* 2003

[45] G. Lu, B. Krishnamachari and C. S. Raghavendra, “Performance Evaluation of the IEEE 802.15.4 MAC for Low-Rate Low-Power Wireless Networks”, *EWCN*, held in conjunction with the IEEE International Performance Computing and Communications Conference (IPCCC), April 2004.

[46] G. Lu, B. Krishnamachari, C. S. Raghavendra, “An Adaptive Energy-Efficient and Low-Latency MAC for Data Gathering in Sensor Networks”, *4th IEEE International Workshop on Algorithms for Wireless, Mobile, Ad Hoc and Sensor Networks WMAN*, April 2004.

[47] G. Lu, B. Krishnamachari, C. S. Raghavendra, “DMAC: An Adaptive Energy-Efficient and Low-Latency MAC for Tree-based Data Gathering in Sensor Networks”, Submitted to Wireless Networks (WINET-Kluwer)

[48] G. Lu, N. Sadagopan and B. Krishnamachari, Ashish Goel, “Delay Efficient Sleep Scheduling in Wireless Sensor Networks”, *IEEE INFOCOM*, Miami, FL, March 2005

[49] G. Lu, N. Sadagopan and B. Krishnamachari, Ashish Goel, “Delay Efficient Sleep Scheduling in Wireless Sensor Networks”, In journal submission

[50] G. Lu, B. Krishnamachari, “Energy Efficient Joint Link Scheduling and Power Control in Wireless Sensor Networks”, *IEEE SECON*, 2005

[51] E. Shih, P. Bahl, M. J. Sinclair, “Wake on wireless:: an event driven energy saving strategy for battery operated devices”, *Proceedings of the 8th annual international conference on Mobile computing and networking*, 2002

[52] X. Yang and N. Vaidya, “A Wakeup Scheme for Sensor Networks: Achieving Balance between Energy Saving and End-to-end Delay”, *Proceedings of the 10th IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS 2004)*, 2004

[53] M. J. Miller and N. H. Vaidya, “Minimizing Energy Consumption in Sensor Networks Using A Wakeup Radio”, *IEEE WCNC*, 2005

[54] C. Schurgers, V. Tsiatsis, S. Ganeriwal an M. Srivastava, “Topology Management for Sensor Networks: Exploiting Latency and Density”, *ACM MobiHoc* 2002

[55] C. Schurgers, V. Tsiatsis, S. Ganeriwal an M. Srivastava, "Optimizing Sensor Networks in the Energy-Latency-Density Design Space", *IEEE Tran. on Mobile Computing* 2002

[56] Jan M. Rabaey *et. al*, "PicoRadio Supports Ad Hoc Ultra-Low Power Wireless Networking", *IEEE Computer Magazine*, 2000

[57] L. Gu and J. A. Stankovic, "Radio-Triggered Wake-Up Capability for Sensor Networks", *Proceeding of Real-Time and Embedded Technology and Applications Symposium (RTAS)* 2004

[58] P. Skraba, H. Aghajan, A. Bahai, "RFID Wake-up in Event Driven Sensor Networks", *Poster in Sigcomm* 2004

[59] M. L. Sichitiu, "Cross-Layer Scheduling for Power Efficiency in Wireless Sensor Networks", *IEEE Infocom* 2004

[60] W. Ye, J. Heidemann, and D. Estrin, "Medium Access Control with Coordinated, Adaptive Sleeping for Wireless Sensor Networks", *IEEE/ACM Transaction on Networking*, To Appear.

[61] A. Woo and D. E. Culler, "A Transmission Control Scheme for Media Access in Sensor Networks", *ACM MobiCom* 2001

[62] R. Zheng, J. C. Hou and L. Sha, "Asynchronous Wakeup For Ad Hoc Networks", *ACM MobiHoc* 2003

[63] R. Zheng, R. Kravets, "On-demand Power Management for Ad Hoc Networks", *IEEE Infocom* 2003

[64] A. El-Hoiydi, "Aloha with Preamble Sampling for Sporadic Traffic in Ad Hoc Wireless Sensor Networks", *roceedings of IEEE International Conference on Communications (ICC)*, April 2002

[65] J. Polastre, J. Hill and D. Culler, "Versatile Low Power Media Access for Wireless Sensor Networks", *ACM Sensys*, 2003

[66] A. El-Hoiydi and J. D. Decotignie, "WiseMAC: An Ultra Low Power MAC Protocol for Mulit-Hop Wireless Sensor Networks", *Proceedings of First Internaltional Workship on Algorithmic Aspects of Wireless Sensor Networks(ALGOSENSORS)*, July 2004

[67] A. K. Das, R. J. Marks, P. Arabshahi, A. Gray, "Power Controlled Minimum Frame Length Scheduling in TDMA Wireless Networks with Sectored Antennas", *IEEE Infocom*, March 2005

[68] M. Zuniga, B. Krishnamachari, "Analyzing the Transitional Region in Low Power Wireless Links", *IEEE SECON*, October 2004

[69] Y. Sankarasubramaniam, O. B. Akan, I. F. Akyildiz, "ESRT: Event-to-Sink Reliable Transport in Wireless Sensor Networks", *ACM MobiHoc* 2003

[70] MICA2 Mote, in *http://www.xbow.com*

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

[72] Power Aware Sensing Tracking and Analysis, *http://pasta.east.isi.edu*.

[73] *802.15.4-2003 IEEE Standard for Information Technology-Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) specifications for Low Rate Wireless Personal Area Networks (LR-WPANS)*, 2003.

[74] L. Bao and J.J. Garcia-Luna-Aceves, "A new approach to channel access scheduling for ad hoc networks", *ACM MobiCom*, 2001

[75] C. Guo, L.C. Zhong, and J.M. Rabaey., "Low power distributed MAC for ad hoc sensor radio networks", *IEEE GlobalCom* 2001

[76] E. Shih, S. Cho, N. Ickes, R. Min, A. Sinha, A. Wang, A. Chandrakasan, "Physical layer Driven Protocol and Algorithm Design for Energy-Efficient Wireless Sensor Networks", *ACM MobiCom* 2001

[77] K. Sohrabi, J. Gao, V. Ailawadhi, and G. J. Pottie, "Protocols for Self-Organization of Wireless Sensor Networks", *IEEE Personal Communication*, Oct. 2000

[78] IEEE 802.11, *802.11-1999 IEEE Standard for Information Technology - LAN/MAN - Specific requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications*, 1999.

[79] A. Woo, D. Culler, "A Transmission Control Scheme for Media Access in Sensor Networks", *Mobicom*, July 2001.

[80] A. Cerpa, N. Busek, and D. Estrin, "SCALE: A tool for Simple Connectivity Assessment in Lossy Environments", CENS Technical Report, September 2003.

[81] J. W. Suurballe, "A Quick Method for Finding Shortest Paris of Disjoint Paths", *Networks*, 14(1974) pp. 125-145

[82] Orinoco WLAN card, http://www.orinoco.com

[83] Z. Abrams, A. Goel, and S. Plotkin, "Set k-cover algorithms for energy efficient monitoring in wireless sensor networks", *ACM/IEEE IPSN* 2004

[84] R. Bhandari, "Optimal physical diversity algorithms and survivable networks", *Proc. Second IEEE Symposium on Computers and Communications* 1997

[85] S. Pattem, B. Krishnamachari, and R. Govindan, "The Impact of Spatial Correlation on Routing with Compression in Wireless Sensor Networks," *ACM/IEEE International Symposium on Information Processing in Sensor Networks (IPSN)*, April 26-27, Berkeley, CA

[86] RIP, "Routing Information Protocol", in RFC 1058 (RFC1058)

[87] H. Gupta, S.R. Das, Q. Gu, "Connected Sensor Cover: Self-Organization of Sensor networks for Efficient Query Execution", *ACM MOBIHOC*, June 2003

[88] C. E. Perkins and E. M. Royer, "Ad-hoc On Demand Distance Vector Routing", *Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications*, New Orleans, LA, February 1999

[89] D. Bertsekas, R. Gallager, Data Networks, Prentice-Hall

[90] JouleTrack, http://carlsberg.mit.edu/JouleTrack/

[91] M. H. A. Ahmed, H. Yanikomeroglu, D. D. Falconer, and S. Mahmoud, "Performance enhancement of joint adaptive modulation, coding and power control using cochannel-interferer assistance and channel reallocation", *IEEE WCNC*, 2003

[92] S. Aldosari and J. Moura, "Fusion in sensor networks with communication constraints", *ACM/IEEE IPSN*, 2004

[93] K. Arisha, M. Youssef, and M. Younis, "Energy-aware TDMA-based MAC for sensor networks", *IEEE Workshop on Integrated Management of Power Aware Communications, Computing, and Networking (IMPACCT)*, 2002

[94] E. Armanious, D. D. Falconer, and H. Yanikomeroglu, "Adaptive modulation, adaptive coding, and power control for fixed cellular broadband wireless systems", *IEEE WCNC*, 2003

[95] G. Barriac, R. Mudumbai, and U. Madhow, "Distributed beamforming for information transfer in sensor networks", *ACM/IEEE IPSN*, 2004

[96] R. Cristescu, B. Beferull-Lozano, and M. Vetterli, “On network correlated data gathering”, *IEEE INFOCOM* 2004

[97] A. E. Gamal, C. Nair, B. Prabhakar, E. Uysal-Biyikoglu, and S. Zahedi, “Energy-efficient scheduling of packet transmissions over wireless networks”, *IEEE INFOCOM* 2002

[98] W. Heinzelman, J. Kulik, and H. Balakrishnan, “Adaptive protocols for information dissemination in wireless sensor networks”, *ACM/IEEE MOBICOM* 1999

[99] G. Holland, N. Vaidya, and P. Bahl, “A rate-adaptive MAC protocol for multi-hop wireless networks”, *ACM/IEEE MOBICOM* 2001

[100] C. Intanagonwiwat, R. Govindan, and D. Estrin, “Directed Diffusion: A scalable and robust communication paradigm for sensor networks”, *ACM MOBICOM* 2000

[101] B. Prabhakar, E. Uysal-Biyikoglu, and A. E. Gamal, “Energy-efficient transmission over a wireless link via lazy packet scheduling”, *IEEE INFOCOM* 2001

[102] C. Schurgers, V. Raghunathan, and M. B. Srivastava, “Modulation scaling for real-time energy aware packet scheduling”, *IEEE GLOBECOM* 2001

[103] R. Min, M. Bhardwaj, S. Cho, A. Sinha, E. Shih, A. Wang, and A. P. Chandrakasan, “Low-power wireless sensor networks”, *14th International Conference on VLSI Design* 2001

[104] G. J. Pottie and W. J. Kaiser, “Wireless integrated network sensors”, *Communications of the ACM*, 43(5):551-558, May 2000

[105] A. Cerpa, J. Elson, D. Estrin, L. Girod, M. Hamilton, and J. Zhao, “Habitat Monitoring: Application Driver for Wireless Communications Technology”, *ACM SIGCOMM Workshop on Data Communications in Latin America and the Caribbean*, April 2001

[106] A. Mainwaring, J. Polastre, R. Szewczyk, D. Culler, and J. Anderson, “Wireless Sensor Networks for Habitat Monitoring”, *IEEE WSNA* 2002