DESIGN CONSIDERATIONS IN

WIRELESS SENSOR NETWORKS

by

Steven A. Borbash

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2004

ACKNOWLEDGEMENTS

Thanks to Susan, who carried a heavy load so that I could do this research.

Thanks also to the National Security Agency, which sponsored me with a nine-month leave to pursue this research, and paid the tuition.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1

Introduction

This dissertation investigates three topics in the design of wireless sensor networks: cross-layer optimization, the feasibility of scheduling as a medium access method, and neighbor discovery.

Sensor networks are large sets of small, inexpensive devices with hardware for sensing and a radio for communication with the other sensors. Sensor networks are being enabled by the convergence of several technologies at once. The advent of cheap, low-power microprocessors, sensor technology, and low-power RF design has made it possible to conceive of large networks which can together do what might be impossible (or too costly) to do with fewer, more expensive nodes.

Wireless sensor networks are "ad hoc" networks, which means that the topology of the network is not planned, but must be decided by the network nodes themselves. Many fundamental questions about wireless ad hoc networks remain unanswered. Among the questions considered in this dissertation are: To what extent is "layered networking" satisfactory for sensor networks? Is scheduling or contention a better way to control medium access in sensor networks? With what power should nodes in the network transmit? Will nodes determine their neighbors, and if so, how?

Ad hoc sensor networks are likely to be designed to fit one application very well. This is because sensors will in many cases face a short lifetime as a result of the limitations of battery technology, and energy can be conserved by removing extraneous functions. The result of applying this principle is that sensor architectures will be optimized for two things: lasting a long time, and doing well the one thing for which they were designed.

If a given wireless sensor network is to have only one or two objectives, there is little reason to maintain the generality of the OSI model ("layered networking" [1]) within the network stack of the sensor. We should optimize lifetime or performance by making connections across layers of the network stack. In a general purpose computer, which must handle a large set of applications and conform to many networking standards, layering is a reasonable response to the great complexity of software. In a sensor which has a single application and no requirement to conform to a standard (at this time), layering serves as an obstacle to performance, lifetime, or both.

For example, some routing algorithms decide which link to forward a packet onto by choosing the path with smallest end-to-end expected delay. A better fit for an energy-constrained node might be to choose the path consuming least total energy. However, energy is a physical layer parameter, which would not be available in the routing calculation if we maintained a strict boundary between the routing, link and physical layers.

It is quite unclear how the optimal communication system should work, absent layering. We largely keep the existing layered model, but allow information to be shared between the layers where there are clear advantages to doing so.

In Chapter 2, we consider cross-layer optimization in the context of a sensor network whose task is event detection. Distributed event detection is a prominent application for sensors. One can imagine for example that sensors might be deployed in a battlefield solely to report that the enemy has been detected, or in some remote location to report that a seismic event has occurred there.

The problem is to minimize the probability of error and maximize lifetime. The solution we provide is a pair of link metrics for the Bellman-Ford algorithm. The metrics capture the notions of residual energy, energy conservation, and the connection between the application and the network.

Contention-based medium access control (MAC), such as that which is in 802.11 wireless LANs, is often preferred to scheduled access to the wireless medium. However, contention-based MACs have well-known disadvantages, including wide variability in delay of transmissions, poor performance in heavily loaded networks, and wasted energy when multiple users attempt to transmit simultaneously. For energy-related metrics typical of sensor networks, scheduling would clearly be superior for mediating access.

For these reasons scheduling deserves a closer analysis for wireless networks. In Chapters 3 and 4 we provide new theoretical results on the scheduling problem in ad hoc wireless networks, and more generally, on "feasible matchings," sets of transmissions which can be simultaneously be successfully received. We assume that transmissions are successfully received if the signal to interference and noise ratio (SINR) exceeds a certain threshold at the receiver, and we allow transmitters to vary their transmit power.

In Chapter 3 we consider the problem of scheduling transmissions to satisfy given link demands, such as might be provided by a routing algorithm. We prove that Minimum Length Scheduling is at least as hard as another problem, MAX-SINR-MATCHING. We provide evidence that MAX-SINR-MATCHING is a hard problem, but also identify a promising heuristic for it. We demonstrate that in certain sub-cases, the scheduling problem is tractable, and for those cases we provide efficient algorithms for computing the minimum schedule length.

The results obtained in Chapter 4 apply to all wireless networks, not only ad hoc or sensor networks. The theorems of this chapter are fundamental results on the SINR model, which has been a standard model for the cellular telephony community since the mid-1990s. At that time, it was discovered that questions about transmission power and the SINR could be cast as matrix eigenvalue problems [2].

Our results exploit the algebraic structure in new ways. We relate the number of feasible matchings to the SINR requirement of the receivers in a wireless network. When the SINR requirement is high ($\theta > 1$), we prove that between a set of transmitters and a set of receivers, each of size k and disjoint, there can be at most one feasible matching out of $k!$ matchings (Theorem 9). When the SINR requirement is sufficiently low ($\theta < \theta_{safe}$) all matchings are feasible. Theorems 13-17 provide bounds on θ_{safe}. The results involve only transmission powers, noise powers, and channel gains.

These results of Chapter 4 can aid scheduling in ad hoc wireless networks. Chapter 3 shows that scheduling problems become easier when all matchings are known to be feasible. For matchings to be guaranteed feasible, it would be sufficient to employ radio systems with a small enough SINR requirement.

Another problem which is important for ad hoc networks is neighbor discovery. There exist many algorithms for scheduling, routing, and topology control which take as input the set of neighbors of each node. How should this information be obtained? In Chapter 5, a neighbor discovery algorithm is analyzed. The algorithm allows every node in a wireless network to learn its neighboring nodes with some probability over a fixed span of time. This algorithm has the attractive property that the nodes need not be synchronized, i.e. they need not agree on a common clock, nor on a starting time for the algorithm. Since global synchronization is expensive to achieve in a large network, the algorithm fills an important need.

Most of the results in this thesis have been published previously. Chapter 2 was presented at the 2004 NATO Workshop on Cross-layer Design Principles [3]. Chapter 3 was presented at WiOpt 2004 [4]. Chapter 4 was summarized in a one-page abstract in the 2004 International Symposium on Information Theory [5]. The neighbor discovery algorithm of Chapter 5 was introduced in a paper published at MobiHoc 2001 [6].

Chapter 2

Cross-layer optimization in a wireless sensor network

2.1 Introduction

Recent literature on sensor network architectures emphasizes that, unlike general-purpose networks, sensor networks have to make efficient use of limited resources in accomplishing their single goal [7]. Since each network aims at a particular focused objective, it is expected that sensor networks with different goals will be designed differently, with features designed specifically for one application.

In this chapter we consider sensor networks whose design goal is event detection. Although event detection is only one of many applications which may be performed by a sensor network, it is a prominent one.

The problem is, how should energy concerns and the desire to accurately detect events with a distributed sensor network be addressed simultaneously? If we treated layers separately, energy would reside at the physical layer, detection would reside at the application layer, and we would fail to capture their relationship. For this reason layering has been called the “original sin of networking” [8]. In this chapter we propose a scheme to exploit the dependence among the physical, routing and application layers.

Expenditure of energy is affected in complicated ways by many layers of a protocol stack. The power dissipated in the radio's amplifier is related nonlinearly to transmission power, which depends on distance to a receiver, which is chosen by the routing protocol; the capacity of a battery can vary based on whether it is accessed in bursts or periodically, which is affected by the medium access protocol; required transmission power depends upon the SINR requirements of the receiver which depends on the coding scheme, which affects the amount of energy we use in signal processing; the routing tells us what neighbor to transmit to, and propagation loss depends on which neighbor we choose. Such tradeoffs produce a highly complicated optimization problem.

The effort in this paper is not to include every possible detail which can complicate the problem, but rather to cut away enough details so that the remaining problem is simple but still retains the essential flavor of cross-layer optimization.

There exists a sizable literature on the problem of decentralized decision making. (References [9, 10, 11] have more than 100 citations.) It is concerned with strategies that a network of sensors may use to decide between a number of hypotheses about the environment, based on their collective observations. One of the nodes in the network is designated a “fusion center” (or sink) responsible for making the ultimate decision on behalf of the network. If the sensors can afford to transmit their entire observation streams to the fusion center, this corresponds to centralized decision making, and it would have the best performance in terms of probability of error. Each peripheral sensor must decide between a finite number of messages to forward, either to a neighbor or directly to the fusion center.

It is assumed in the literature of decentralized decision making that an error-free communication channel exists, over which sensors communicate. A strategy for the sensors (consisting of local decisions about what to transmit given what has been sensed locally and what has been received from neighbors, and a decision rule for the fusion center) is judged by the accuracy of the sink's decision. It turns out to be very difficult in general to construct optimal decision rules. However there are results that indicate that (a) when observations are conditionally independent, optimal decision rules are likelihood ratio tests [10], and (b) if all the sensors use identical decision rules, accuracy may not suffer much [12, 13, 14].

Since it is unclear how optimal decision rules can be computed in a centralized way, let alone in a distributed environment, we will assume that all sensors use certain identical likelihood ratio tests as decision rules. We expect our tests to have suboptimal error performance [15], but they would be easy to actually implement. By this simplifying assumption, we free ourselves to consider energy performance, which lies at the heart of the practical problems confronting sensor networks.

Specifically, to sustain the assumption of error-free communications, we require a certain signal-to-noise ratio (SNR) be achieved at any receiving station. We have a path loss between each pair of nodes. This leads us to try to organize the nodes into a tree rooted at the sink, with two goals: to ensure the sink receives messages that enable accurate decisions, and to use little energy so that the network can operate for a long time.

Our model accounts for energy spent in the transmission and reception of bits. However, it would be easy to incorporate other energy costs, such as the one associated with sensation, and the processing required for routing and medium access, into our model. Here we focus on the case where communication costs are dominant.

Our contribution is the combination of distributed detection with energy concerns which come together in routing. Specifically, we propose routing metrics for use with the Bellman-Ford algorithm which capture the notions of residual energy, routing diversity, and energy conservation. As we shall show, routing diversity affects the network's error performance. Residual energy and energy conservation affect the network's lifetime.

Distributed detection is examined with energy concerns, but without routing, in [16]. Much work has combined routing with energy concerns but without concern for the detection problem. Two examples are [17, 18].

There are many ways to route and aggregate data. We examine only two which seem natural. The comparisons of this chapter are an exploration of interlayer dependencies, and we do not pretend that our models are optimal.

The paper is organized as follows. There is a detailed description of the network and the detection problem in Section 2.2. Routing metrics and methods of aggregating data are proposed to simultaneously address energy and accuracy concerns. In Section 2.3, results of simulation of the models are presented. In Section 2.4 the simulation results are connected with a recent theoretical result in distributed detection. We conclude in Section 2.5.

2.2 Network model

Let there be a network of N nodes, in which $S_1, \ldots, S_{N-1}$ are sensors and S_0 is a sink (or fusion center). The network operates over some span of time, divided into sensation periods of fixed duration. During each sensation period, each sensor S_i observes a binary valued random variable $Y_i \in \{0, 1\}$. The sink does not make an observation.

There are two possible states of nature during any sensation period, represented by a random variable H which takes values 0 or 1. H takes value 1 with known probability. The variables Y_i are conditionally independent given H. The probabilities $Pr(Y \mid H)$ are assumed to be known.

After a sequence of transmissions, governed by rules to be described below, the sink node will receive one message from each sensor which is an immediate predecessor of it, in a certain tree. The sink will then form a decision $U_0 \in \{0, 1\}$ of the network. If $U_0 = H$ the decision is correct, otherwise it is not. There are costs C_M and C_F associated with misses $(H = 1, U_0 = 0)$ and false alarms $(H = 0, U_0 = 1)$, and the sink node makes a decision which minimizes expected costs. We assume that $C_M = C_F$, allowing us to measure performance as the observed fraction of time that $U_0 = H$.

Nodes may transmit with power no greater than P_{max}. Every receiver is subject to average noise power ν. For S_i to be able to successfully transmit bits to S_j, S_i must be able to reach S_j with $SNR \geq \theta$ at P_{max}. This corresponds to a path loss requirement

$$\alpha(i, j) \geq \theta\nu/P_{max}. \tag{2.1}$$

In this equation, $\alpha(i, j) \in (0, 1]$ represents the fraction of power transmitted by S_i that is received by S_j. We assume that S_i is aware of all nodes S_j satisfying the requirement, perhaps through some neighbor discovery mechanism, such as that presented in Chapter 5.

Next we present two models of data aggregation and routing metrics to go with them. Our co-design of the application and the routing mechanism is part of our cross-layer optimization.

2.2.1 Data aggregation in model M_1

In model M_1, each sensor S_i will have its own observation Y_i as well as one bit from each node which immediately precedes it in the routing tree. S_i computes the majority vote of these input bits, with a fair coin flip in case of ties, to produce its own output U_i. The sink node has no observation of its own, and unlike the sensor nodes, makes a minimum-cost decision described above, based on the one-bit votes which arrive.

With conditionally independent identically distributed observations, as we assume, the optimal decision rules are known to be k-out-of-n rules, of which majority vote is a special case [9]. However it is unclear what the k's ought to be, let alone how to compute these in a distributed setting. Therefore we specify majority vote.

The sink does not know what the routing tree is, or whether any of the nodes in the network have died. Whatever messages (bits) arrive at the sink, they are weighted equally by the sink and a minimum expected cost decision U_0 is made.

2.2.2 Data aggregation in model M_2

In model M_2 each sensor will have its own observation Y_i as well as $k \geq 0$ messages from its immediate predecessors. In M_2 a message from a sensor consists of two integers, $< Z, O >$. The first integer is the number of zeroes observed in the entire subtree of nodes which precede the node; the second integer is the number of ones in that subtree. What node i forwards is $< Z_i, O_i >$ where

$$Z_i = \sum_{j=1}^{k} Z_j + \delta_{Y_i,0}, \tag{2.2}$$

$$O_i = \sum_{j=1}^{k} O_j + \delta_{Y_i,1}. \tag{2.3}$$

What is being forwarded by S_i is a sufficient statistic for all the observations in the subtree rooted at S_i. We do not assume that every node would immediately become aware that some node in its subtree had died; therefore both integers are sent. Unlike in M_1, in M_2 the nodes perform lossless compression. As long as all the nodes can report their data to a successor, the sink can make a centralized decision.

In each model, messages are assumed to be sent in packets whose overhead (for addressing, error control, data labeling, etc.) is a fixed number of bits, equal for each model. Communications are assumed error-free. In practice, schemes for acknowledgments and re-transmissions would need to be specified. The method of wireless medium access must also be specified. Certainly these would affect the energy usage of the networks under study. Also, in order to perform the data aggregation as specified above, each node will have to wait for messages from all immediate predecessors to arrive before computing and transmitting its output message, which could mean that the decisions are delayed, i.e. that the messages regarding sensing period m are still being transmitted during sensing period $m + k$. These important factors are not considered here, so that we may focus on the interaction between the routing, the application, and the energy consumption.

2.2.3 Routing

We assume that all the nodes perform the Bellman-Ford routing algorithm. Routes produced by this distributed algorithm are free of loops. Because there is one destination for all packets (S_0), the routes provided by Bellman-Ford are trees oriented toward the sink node. This justifies the term "routing tree" used throughout the paper.

To perform the Bellman-Ford algorithm, each link in the network must be assigned a cost. These costs are:

$$C_1(i,j) = \frac{I_j}{B_i} \tag{2.4}$$

for M_1 and

$$C_2(i,j) = \frac{P_{ij}}{B_i} \tag{2.5}$$

for M_2, where I_i is the number of nodes who can reach node i with SNR $\geq \theta$

using P_{max}, B_i is the battery level of node i, and $P_{ij} = \theta\nu/\alpha(i,j)$ is the minimum transmit power required for S_i to achieve θ at S_j when the noise power at S_j is the (network-wide constant) ν and the path loss from S_i to S_j equals $\alpha(i,j)$.

The routing metrics are important, because over time (discrete time as measured in sensation periods) the battery levels of the nodes will change, which affects the denominators of C_1 and C_2. Nodes may die, which affects the numerator of C_1. As these changes occur, the routing tree adapts. In Sections 2.2.4 and 2.2.5 we discuss the reasoning behind C_1 and C_2. In Section 2.2.6 we discuss the denominators of C_1 and C_2.

2.2.4 The link cost for M_1

The numerator, I_j, of the cost metric used in Bellman-Ford routing in M_1 will now be discussed. Essentially, it represents a desire for the routing algorithm to help the sensor network achieve its goal of accuracy. This is an example of cross-layer optimization.

Consider the topology of Figure 2.1 which results from a particular routing choice. This topology will tend to create errors, because under the forwarding rules of M_1, node R in the tandem portion of the network performs a majority vote of one bit input from its immediate predecessor (which is likely to be correct), and its one bit observation Y_R. If the observation $Y_R \neq H$ node R simply performs a coin flip to decide U_R. The same difficulty exists with U_S. Therefore, the routing algorithm for M_1 should aim to avoid such topologies.

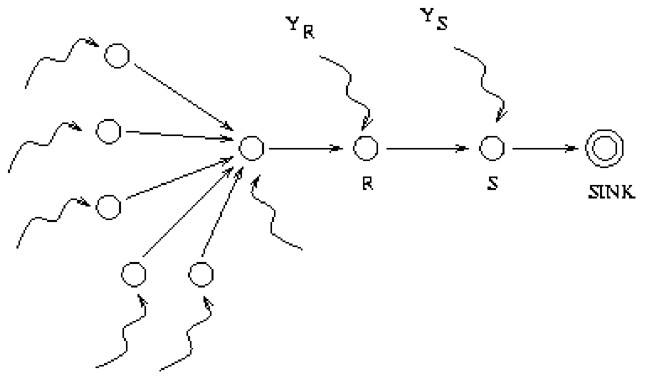


Figure 2.1: Inappropriate routing for the data aggregation method of M_1. Curved arrows represent observations.

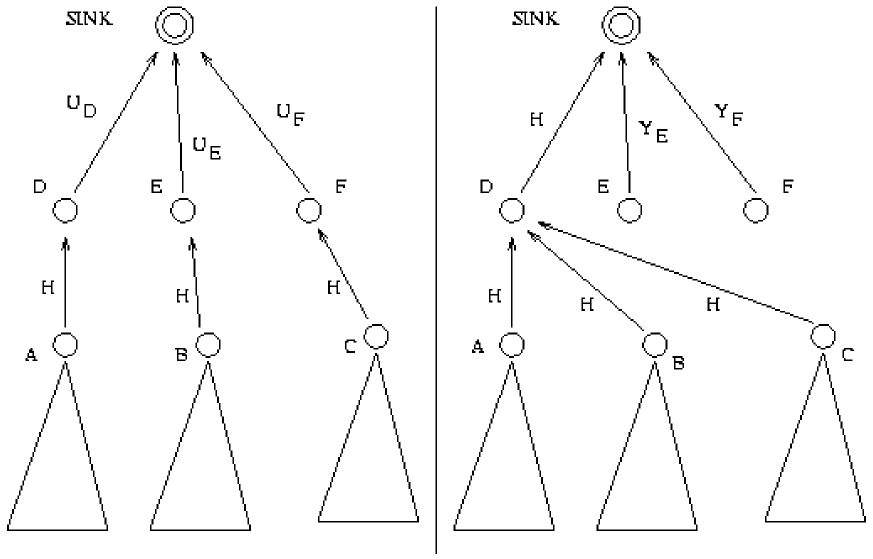


Figure 2.2: If each node is to forward a single bit, as in M_1, the routing on the left will produce a better final decision than the the routing on the right.

Consider the example network depicted in Figure 2.2. Assume that $Pr(H = 1) = 1/2$ and $Pr(Y_i = H) = 2/3$ at every sensor i. The triangles represent large subtrees of nodes whose observations are aggregated into single bits at nodes A, B, and C. For simplicity assume that the weight of evidence at these nodes is so great that $U_A = U_B = U_C = H$. On the left hand side of Figure 2.2, each of U_D, U_E and U_F equal H about 5/6 of the time, so the sink's decision will be wrong 2/27 of the time. On the right hand side of Figure 2.2, all the evidence is concentrated into U_D, which will always equal H, but $U_E = Y_E$ and $U_F = Y_F$ so the sink's decision will be wrong 3/27 of the time. Thus, the more balanced tree, with approximately equal bushiness at each layer, seems to perform better.

This argument for "route diversity," by which we mean that data streams do not coalesce as they approach the sink, can be applied recursively to the triangles feeding A, B and C.

The factor I_j in C_1 encourages route diversity. To achieve diversity each node should transmit to nodes that other nodes cannot reach. Consider Figure 2.3, in which the possible predecessors of D, E and F are $\{A, B, C\}$, $\{B, C\}$, and $\{C\}$ respectively. If we chose routing with the metric of M_2, we would get the undesirable routes of Figure 2.4. To spread out the traffic, we seek a "system of distinct representatives" (SDR) [19] of these three sets, namely A for $\{A, B, C\}$, B for $\{B, C\}$, and C for $\{C\}$. While performing a matching algorithm to find an SDR is unrealistic in a distributed setting, we can approximate the effect by having each node favor

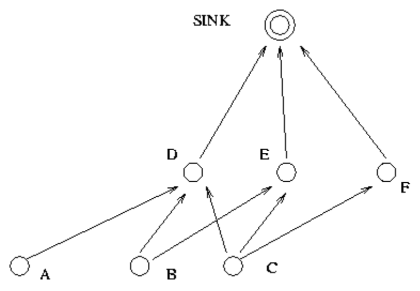


Figure 2.3: Example. Arrows indicate possible next hops.

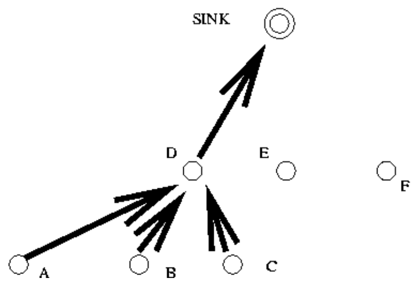


Figure 2.4: Undesirable routing tree for network of Figure 2.3 if the data aggregation method of M_1 is used.

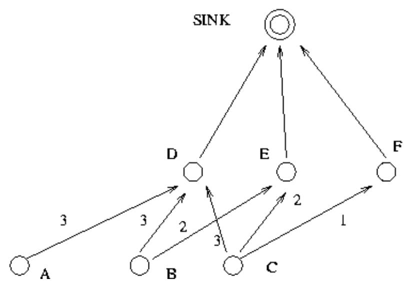


Figure 2.5: Letting cost of link (i, j) equal the number of arrows pointing into j.

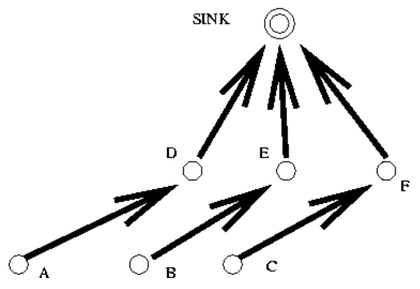


Figure 2.6: Routing tree for network of Figure 2.3 using the link costs of Figure 2.5.

successors which have fewer possible predecessors. For example C has 1 possible predecessor, B has 2 and A has 3. So we set the costs for these links as in Figure 2.5 and get the desirable routing of Figure 2.6.

2.2.5 The link cost for M_2

The numerator of the link cost for M_2, P_{ij}, is proportional to the power required to transmit a bit on link (i, j) such that it arrives at the receiver with sufficient

SNR. Thus, in M_2, unlike in M_1, there is a clean separation between the purpose of routing (energy conservation) and the purpose of the data aggregation (to achieve accurate decisions at the sink).

2.2.6 Taking account of finite energy

As has been discussed elsewhere [8], there is a crucial difference between problems where energy is renewable, and treated simply as a cost, and where energy is a finite non-renewable resource. When energy is only a cost it is typical to treat it the same way over time. But when running out of energy means a node dies, it may be wise to be more conservative as it dwindles.

For both models M_1 and M_2, therefore, the cost for link (i, j) was divided by the battery capacity of S_i. Thus when S_i had relatively little energy the cost of all paths going through S_i appeared higher.

Because this is an investigation of heuristics, we decided also to investigate link metrics identical to the ones above but with the denominators replaced by B_j, so that link cost would be high when a receiver had little energy, as opposed to the above where link costs are high when the transmitter has little energy. Thus we define M_3 and M_4 to aggregate data as do M_1 and M_2 respectively, and to use link metrics:

$$C_3(i,j) = \frac{I_j}{B_j} \tag{2.6}$$

for M_3 and

$$C_4(i,j) = \frac{P_{ij}}{B_j} \tag{2.7}$$

for M_4.

Example Consider the 10 nodes placed as in Figure 2.7. Neighbor relations between the 10 nodes are indicated by an edge connecting them. The edge label indicates the power required at the transmitting node to achieve θ at the receiving node; in this case $P_{max} = 3000$. Node 1 is the sink. Suppose the residual battery levels at nodes $1, 2, \ldots, 10$ are 10000, 20000,..., 100000 respectively. Then the Bellman-Ford routing trees computed with the link metrics from Equations (2.4), (2.5), (2.6), (2.7) are shown in Figure 2.8.

2.3 Simulation results

Each simulation trial consists of the following steps.

1. N nodes are randomly placed in a square area. There is no distinguished place for the sink node; it may be near a border or near the center. Based on these random two-dimensional placements, P_{max}, ν, and a path loss which is proportional to distance cubed between nodes, each node's neighbors are enumerated, together with the path losses to them. (The squares are sized and the constants are chosen so that the average number of neighbors per node is about 4.)

2. A sufficiently large dataset, consisting of values of H and $Y_1, \ldots, Y_{N-1}$ for each sensing period are generated.

3. Equal battery capacities are given to each of the N nodes.

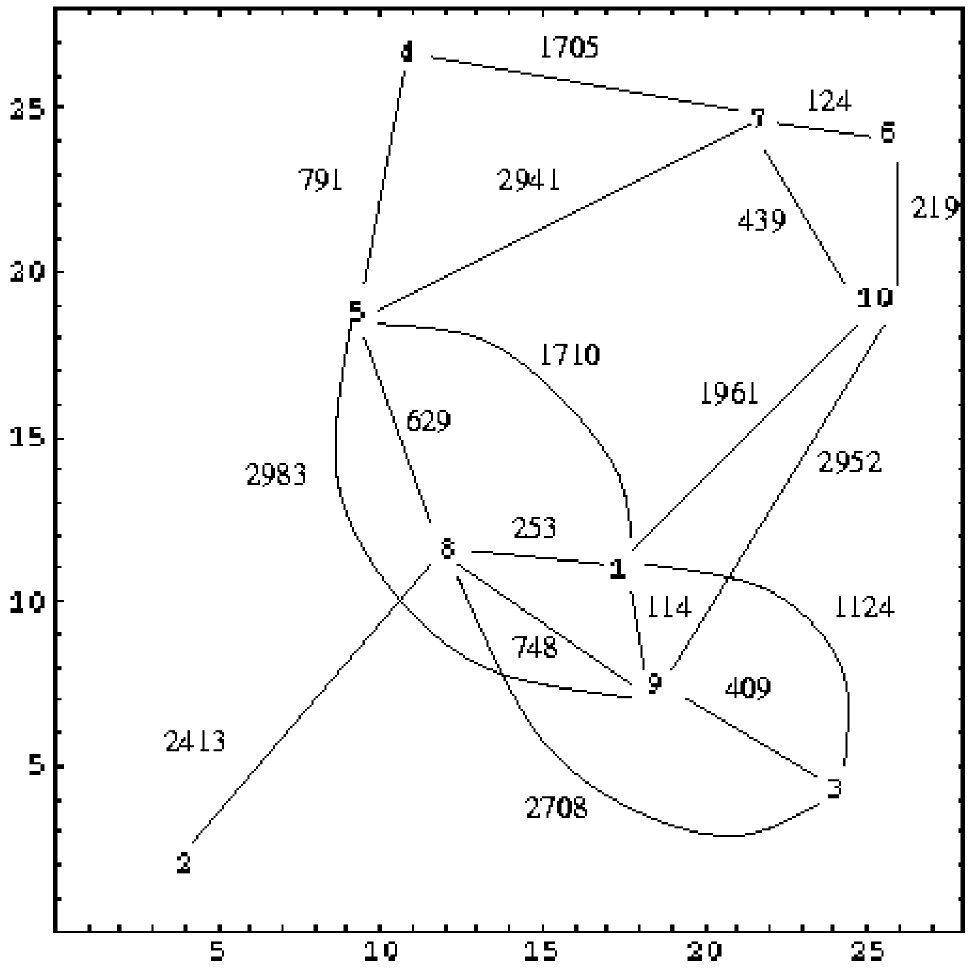


Figure 2.7: Ten node graph with minimum transmit powers. Node 1 is the sink. For ease of display, the directed graph is here represented as undirected.

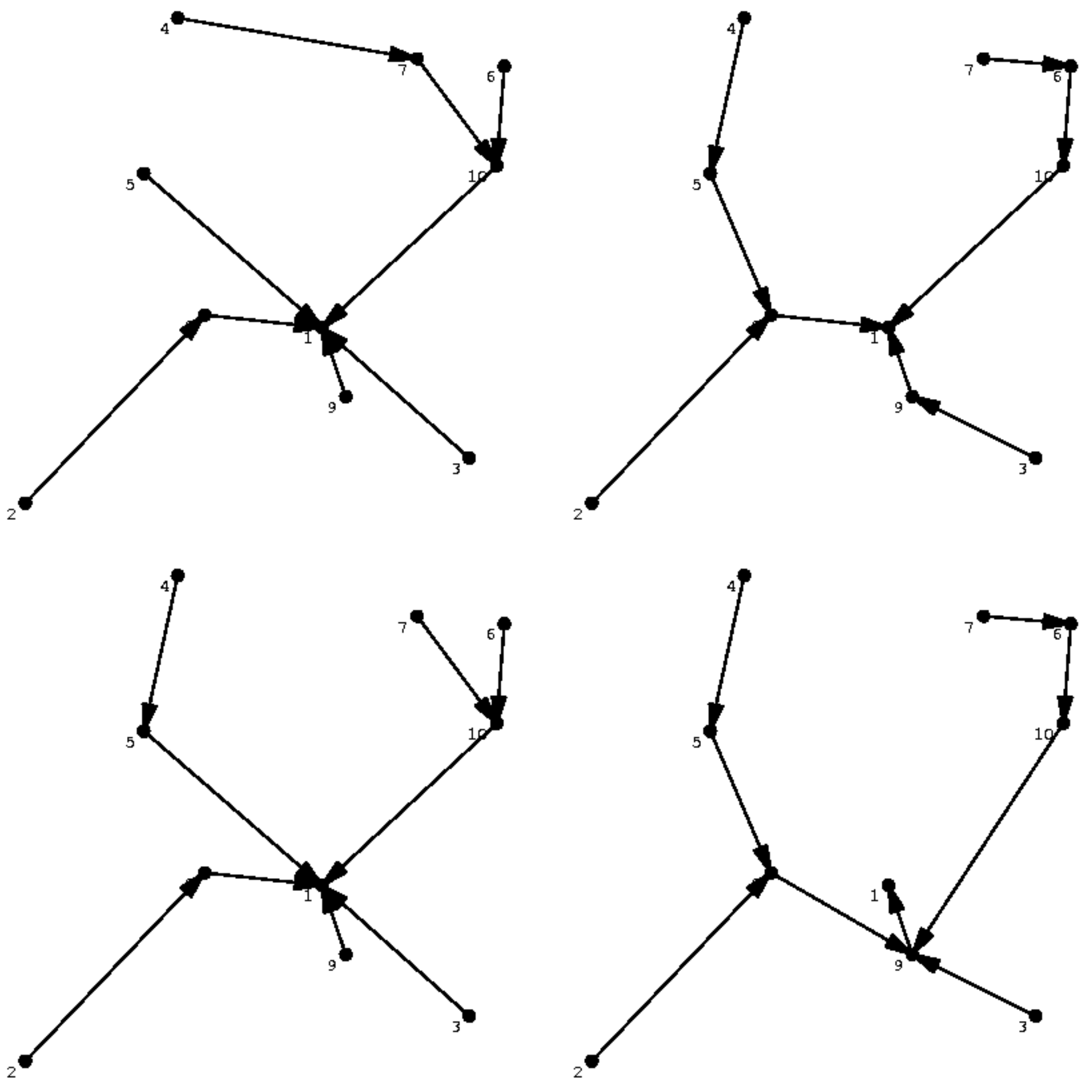


Figure 2.8: Routing trees computed by Bellman-Ford for the network depicted in Figure 2.7 with nodes $1, 2, \ldots, 10$ having battery levels 10000, 20000, …, 100000 respectively. Using M_1 gives the routing tree in the upper left; M_2, upper right; M_3, lower left; M_4, lower right.

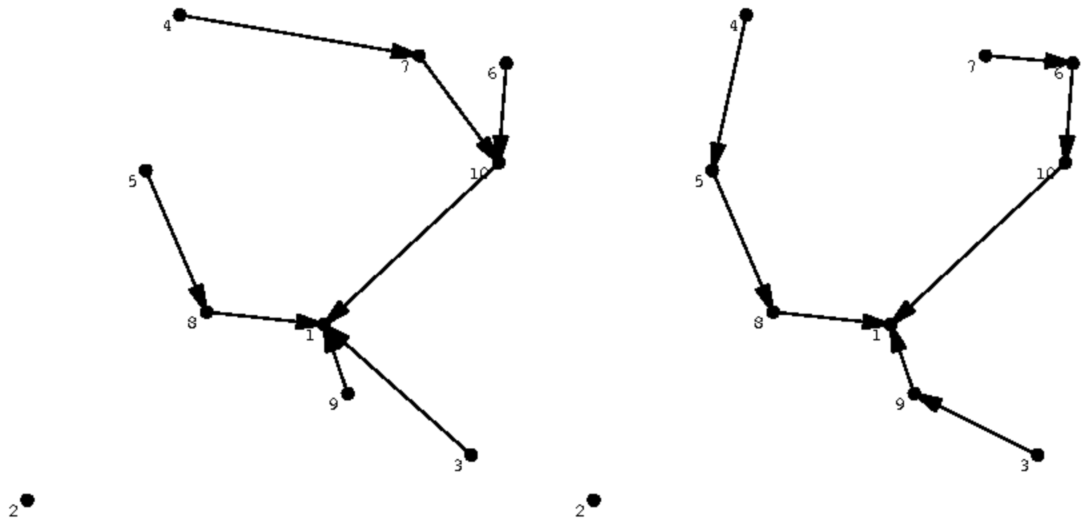


Figure 2.9: The routing changes over time. Starting from the routing and battery capacities of Figure 2.8, we show the first changes in routing for M_1 (left) and M_2 (right). In M_1 nine sensation periods pass before node 2 has insufficient energy to exceed the SNR at any neighbor with a full message, so he drops out. The new routes are shown. In M_2, three sensation periods pass before node 2 drops out. The new routes are shown.

4. Model M_1 is simulated with the losses, data, and battery capacities. The simulation stops when fewer than 1/2 the nodes are alive. A node is "alive" when it (a) has enough energy to transmit to its next-hop (as dictated by the routing algorithm) and (b) all the nodes along some path to the sink, through the next-hop, have enough energy to transmit to their successors.

5. The battery capacities are reset and other models (M_2, M_3 and M_4) are simulated on the same losses, data and battery capacities.

An advantage of Bellman-Ford is that it can run independently of the sensing and transmission of reports, and continually track the ideal routing as link metrics change and new advertisement messages are received. In practice the distributed Bellman-Ford algorithm does not immediately converge: changes to link costs must propagate over the whole network, and until they have, routes may be stale and packets may be forwarded to nodes with insufficient energy [20]. In order to focus on the differences between the routing metrics, we assumed that each node had obtained up-to-date information at the time it determines its next hop.

We also chose not to charge energy for the running of the Bellman-Ford algorithm. Although certainly there is a cost in practice, since all our models use Bellman-Ford, all would pay the same price, and our comparisons below would remain qualitatively valid.

We now present our simulation results in subsections.

Table 2.1: Comparing M_1 to M_2, assuming $Pr(H = 1) = 1/2$.

	number of nodes (incl. sink)					
$Pr(Y_i = H)$	$N = 10$		$N = 20$		$N = 40$	
0.55	0.58	43.7	0.57	22.0	0.62	14.3
	0.60	32.7	0.61	9.4	0.65	4.3
0.6	0.65	41.9	0.68	22.8	0.69	13.3
	0.69	27.9	0.75	10.4	0.77	4.3
0.7	0.78	46.7	0.83	23.2	0.81	13.7
	0.82	34.7	0.90	10.4	0.90	4.3
0.8	0.87	40.0	0.91	23.0	0.92	14.4
	0.92	28.9	0.94	10.5	0.95	4.7
0.9	0.96	47.2	0.971	23.0	0.956	15.5
	0.97	31.7	0.970	11.0	0.964	4.9

2.3.1 Comparing M_1 to M_2

Each cell in Table 2.1 contains an entry of the form

$$\begin{array}{cc} M_1\text{'s accuracy} & M_1\text{'s lifetime} \\ M_2\text{'s accuracy} & M_2\text{'s lifetime} \end{array} \tag{2.8}$$

representing averages from at least 200 independent runs. The lifetime equals the (average) number of sensation periods before half the nodes die. The accuracy equals the (average) fraction of times that $U_0 = H$ before half the nodes die.

By comparing the columns in the table, we see that an increasing number of nodes causes a decrease in the life of the network. (Remember that unless otherwise specified, the number of neighboring nodes is being kept at about 4; this means that when we go from $N = 10$ to $N = 20$ nodes, the 20 nodes occupy a larger area than do the 10. Changes in density are considered below.)

By comparing the rows in the table, we see that as individual nodes have better sensors, the network's accuracy improves.

In comparing M_1 to M_2, the former always has a longer lifetime and the latter always has a higher accuracy, except in the $N = 20, Pr(Y_i = H) = 0.9$ case.

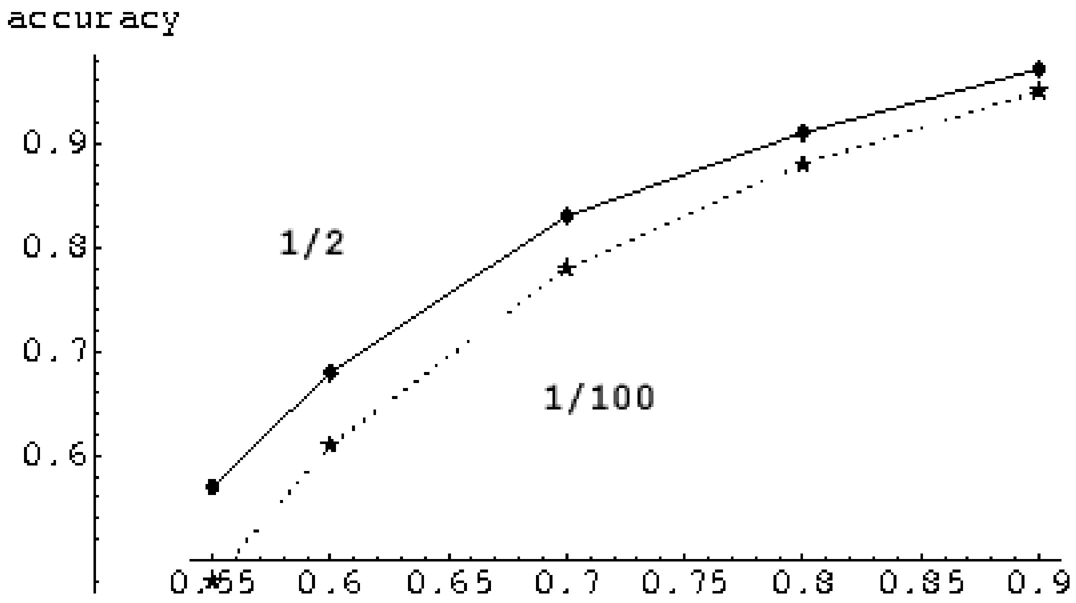
2.3.2 Effect of unequal priors

To see the effect when $Pr(H = 1) \neq 1/2$ we ran simulations under the same conditions as above, but with $Pr(H = 1) = 1/100$. Note that the data aggregation rules of M_1 and M_2 have not changed; e.g., in M_1 each node still takes a majority vote. However the sink node's decision U_0 does depend on $Pr(H = 1)$. Each cell in Table 2.2 is based on at least 100 independent runs. Its entries have the form described in (2.8).

Changing the prior from 1/2 to 1/100 (or away from 1/2 in either direction, we conjecture) hurts the accuracy of M_1, but doesn't affect M_2, as shown in Figure 2.10. The effect is probably a result of the majority vote rule being most appropriate when $Pr(H = 1) = 1/2$ and less appropriate otherwise. Changing the prior has no consistent effect on lifetime for either M_1 or M_2.

Table 2.2: Comparison of M_1 to M_2, assuming $Pr(H = 1) = 1/100$.

	number of nodes (including sink)					
$Pr(Y = H)$	$N = 10$		$N = 20$		$N = 40$	
0.55	0.48	48.1	0.48	23.5	0.47	16.6
	0.53	31.6	0.57	10.8	0.57	4.9
0.6	0.56	43.0	0.61	23.7	0.58	13.4
	0.65	30.3	0.73	11.6	0.75	4.0
0.7	0.75	45.1	0.78	24.2	0.81	15.3
	0.82	34.0	0.89	10.7	0.89	4.7
0.8	0.86	45.4	0.88	22.5	0.88	15.0
	0.94	28.2	0.94	10.6	0.95	4.8
0.9	0.94	49.5	0.95	23.8	0.96	13.6
	0.97	29.1	0.98	10.1	0.98	4.4



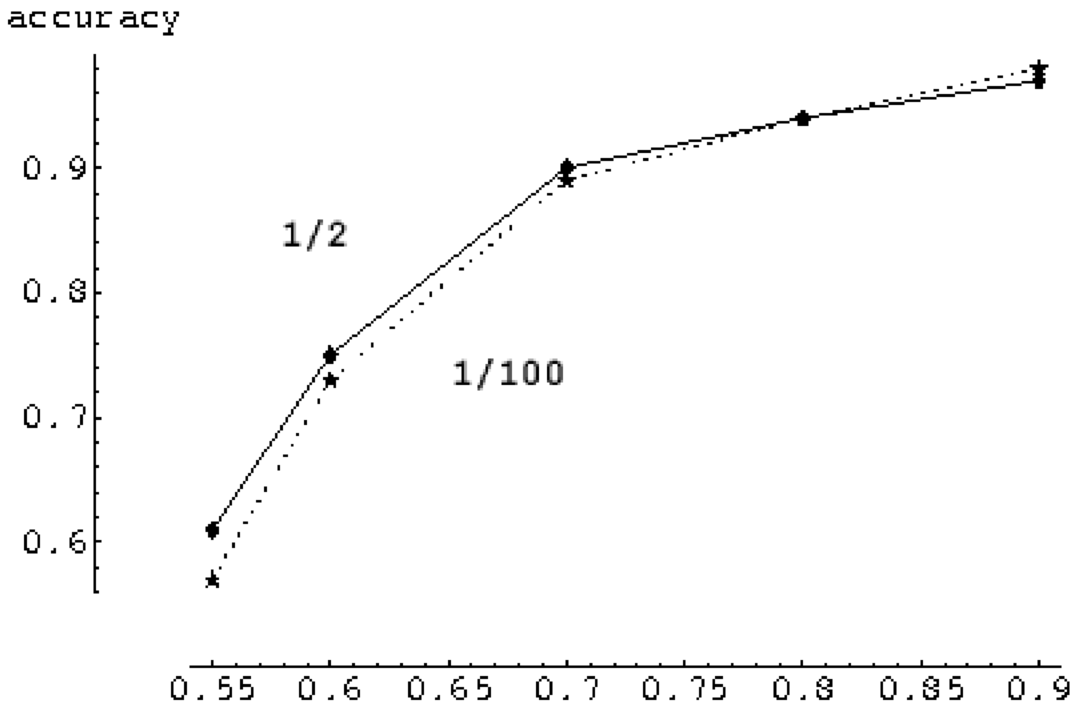


Figure 2.10: Comparison of network accuracy when $Pr(H = 1) = 1/2$ and $Pr(H = 1) = 1/100$ for M_1 (top) and for M_2 (bottom). Based on simulations with $N = 20$ nodes.

2.3.3 Comparison of M_1, M_2, M_3 and M_4

We can see from Figure 2.11 that M_1 and M_3 act almost identically, as do M_2 and M_4. A closer look at the data shows that M_1 and M_3 have slightly better energy performances than M_2 and M_4 respectively, with accuracies between them being close in all cases.

2.3.4 Effect of network density

All the above simulations used an area such that the average number of neighboring nodes would be 4. We decreased the area so that the expected number of neighbors would be 8. We assume $Pr(H = 1) = 1/2$. Each point in Figure 2.12 is based on 50 independent runs.

It has been observed elsewhere that increased density improves network lifetime, because it reduces the average transmission range, and increases the number of available routes [18]. We see that it also improves the accuracy of the sink's decisions. We suspect that more nodes can communicate to the sink without intermediaries in a denser network, which reduces the probability of error.

2.3.5 Summary of simulation results

We present a summary of the simulation results.

1. As expected, M_1 had a longer lifetime than M_2, because it transmits fewer bits. For the same reason, M_1's accuracy is lower. The accuracy of M_2 is equivalent to a centralized system containing all the currently live nodes.

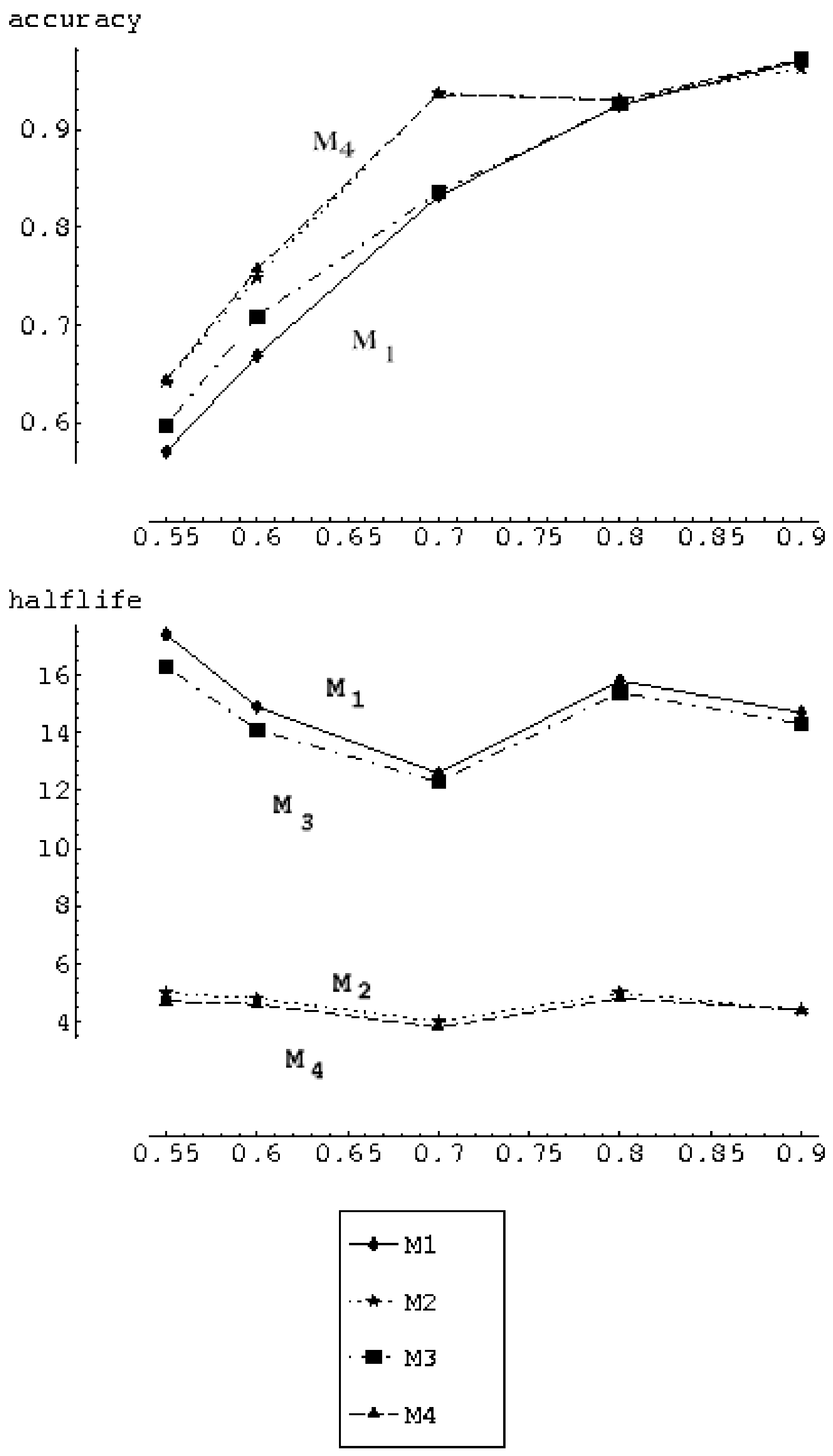


Figure 2.11: Plots of accuracy and lifetime for models M_1, M_2, M_3 and M_4 in a $N = 40$ node network with $Pr(H = 1) = 1/2$. The horizontal axes measure $Pr(Y_i = H)$ for the sensors.

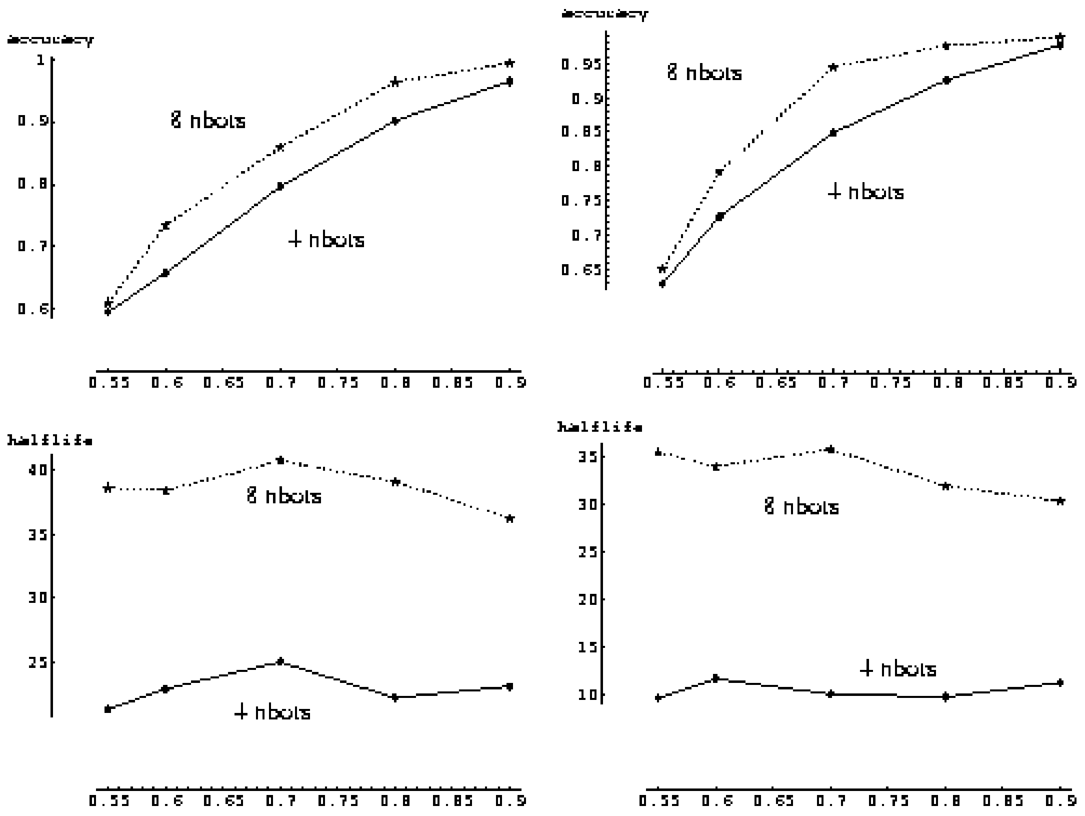


Figure 2.12: Effects of increasing density so that number of neighbors goes from 4 to 8 on average, in a network with $N = 20$ nodes. Top left, effect on accuracy in M_1; top right, effect on accuracy in M_2; bottom left, effect on lifetime in M_1; bottom right, effect on lifetime in M_2.

2. A denser network (more neighbors per node) enjoyed a greater lifetime. This is in part because of the savings in transmission costs. But the increase in connectivity also plays a role, because more routes are available.

3. A denser network is more accurate with all the models we used.

4. Increasing the number of nodes (while keeping density constant) did not improve accuracy, but it did worsen lifetime. Therefore, if nodes are added, they can more profitably be added to the same area the others are in, which would

increase density.

5. Unequal priors ($Pr(H = 1) \neq 1/2$) hurt the accuracy of M_1 slightly, but have no effect on M_2's accuracy and do not affect lifetime.

6. M_1 and M_2 have slightly longer lifetimes than M_3 and M_4 with accuracies nearly equal.

2.4 Relation to a theoretical result

In [21], Shi, Sun and Wesel study the level of quantization sensors should perform. Although they do not consider energy or routing, and restrict attention to one-hop networks, their results are related to ours. The sensors each observe real numbers $H + \nu_i$ where $H \in \{-1, +1\}$ and the ν_i are independent Gaussian noises with mean 0 and variance 1. They send either a single bit or an infinite precision measurement to the sink. In other words, the sensors do not communicate with each other, but only to the sink. See Figure 2.4. The authors report that "it takes fewer than twice as many sensors transmitting a single bit to give the performance of infinite precision sensors." In our terms, M_1 with 40 sensors should perform as well as M_2 with 20 sensors.

The accuracies of M_1 with 40 nodes and M_2 with 20 nodes, with $Pr(H = 1) = 1/2$, are shown in Figure 2.4. (Data come from Table 2.1.) We can see that M_1 still has worse accuracy than M_2 even with twice as many messages. This is in part because because our network is multihop. The multihop network reduces the ability of the sink to correctly weight its inputs, and sensor nodes in our multihop network may be marring the final decision with poor local decisions. Also in [21], all sensors are assumed to be using an identical threshold which is centrally computed to be optimal under the condition of Gaussian noise. Our threshold in M_1 for all nodes is simply half the number of input bits (majority vote). It would be interesting to find a rule to replace the majority vote which would improve the performance of M_1.

2.5 Conclusion

We considered a sensor network whose goals were event detection and long lifetime. We proposed two models of network operation, in which we defined (a) how the nodes would aggregate their observations and (b) deliver data to the sink node.

In M_2 we essentially separated these two concerns, allowing data aggregation to be independent of routing, and used a routing metric which took account of energy concerns. In M_1 we designed the data aggregation method to save energy, by reducing message sizes to a single bit. We observed that there were many ways to

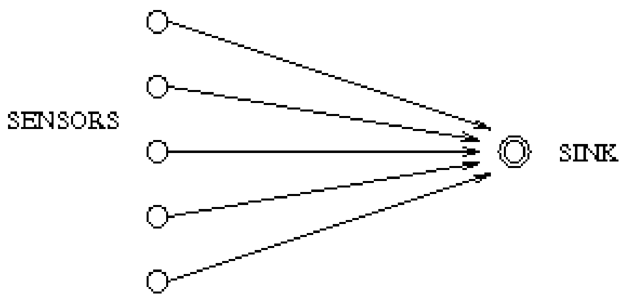
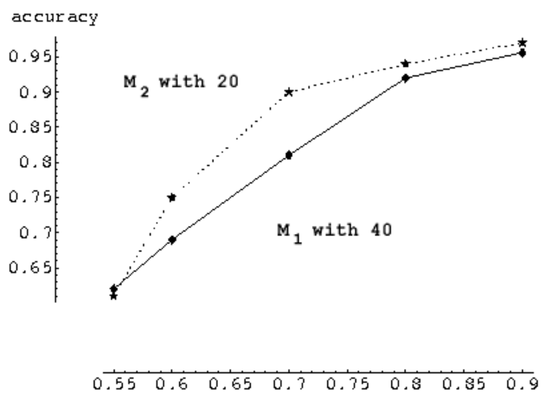


Figure 2.13: One hop network for result of Shi, Sun and Wesel.



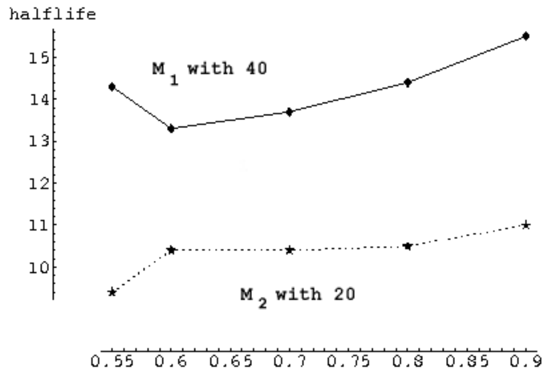


Figure 2.14: Comparing M_1 with 40 nodes to M_2 with 20 nodes. The accuracy of M_2 is better, but the lifetime of M_1 is longer.

route poorly given this data aggregation scheme, and chose a routing metric which encouraged route diversity, which appears to interact well with the application.

Our two routing metrics thus capture aspects of energy efficiency, routing diversity, and residual energy. The routing metrics are a convenient place to perform cross-layer optimization.

The simulation results confirm that M_1 has longer lifetime but less accuracy than M_2. We recalled a theoretical result which claimed that the accuracy of a distributed detection network using single-bit reporting (as our M_1 does) ought roughly to equal the accuracy of a network using infinite-precision reporting (as our M_2 does). We found that our M_1 did not quite meet this standard. However, the theoretical result did not take into account the multi-hop nature of our network, and this may explain the difference.

In any case, without the cross-layer optimization scheme employed in M_1, it is not clear how to achieve gains near what the theory says are possible.

Chapter 3

Wireless link scheduling with SINR constraints and power control

3.1 Introduction

Scheduling is important in wireless networks for at least two reasons. First, a schedule of minimum length provides an upper bound on the network's throughput. Second, scheduling is necessary to avoid collisions. Collisions cost energy, making them undesirable in wireless networks whose nodes have limited energy.

To produce good schedules we need large activation sets, sets of links which can be used concurrently. We call these "feasible matchings." The larger a matching, the greater the parallelism, and the shorter the schedule. In this work we provide new theorems about matchings.

There is a rich literature in the scheduling of wireless networks [23, 24, 25, 26, 27]. Scheduling includes broadcast scheduling and link scheduling. In this paper we consider link scheduling.

Most existing work on wireless link scheduling attempts to model the network's interference constraints as a graph. A matching in the network is identified with a matching of edges in the graph. A sampling of the many examples of this work is [23, 24, 25].

We model interference using the signal to interference and noise ratio (SINR) at the receiving stations. The SINR model was analyzed in the 1990s by researchers in cellular telephony [2, 28, 29, 30]. In [2] it was shown that the question of whether the set of links $T_1 \to R_1, \ldots, T_k \to R_k$, with all $2k$ stations distinct, can simultaneously be activated is a matrix eigenvalue question. One computes the Perron (largest) eigenvalue of a certain nonnegative $k \times k$ matrix. If the eigenvalue is small enough, the set of links can simultaneously be activated with all receivers having adequate SINR, provided we use a power vector that is the associated eigenvector of the matrix.

The SINR model was also used in [26], where average transmission power was minimized subject to an average data rate constraint.

Considerable attention has been given to this model in cellular telephony. In those applications the transmitters are base stations, and the receivers are mobiles. (Or vice versa.) Physical placement of base stations is of course part of the cellular design process.

In ad hoc networks, physical placement may not be under design control.

Also, every node may be required to act as transmitter or receiver, for example to carry transit traffic as part of a routing algorithm. We may desire to produce a schedule during which arbitrary link demands must be met. In this case it is possible to directly apply Theorem 1 or 2 to determine which matchings are feasible. However, we aim to say more about the feasibility of matchings without performing an eigenvalue computation for each matching. Our ability to do this will rest on the same theory of nonnegative matrices that was the basis for Grandhi's result.

In [23], Hajek and Sasaki produce a centralized, strongly polynomial time algorithm for the problem of finding a minimum length schedule among some wireless nodes, such that a set of given link traffic requirements were satisfied. The input included an undirected graph. Two links could simultaneously be active in the schedule if the corresponding edges of the graph were independent, i.e. had no vertices in common. The ability to find minimum length schedules depended then upon finding matchings in a non-bipartite graph, a problem on which there exists considerable graph-theoretic literature.

Their graph represents a wireless network in which one transmission is independent of another. Interference has no effect; the only constraints are that no node can simultaneously transmit and receive, and no node can transmit to or receive from more than one node at a time. In many wireless networks, however, interference remains an impediment even when CDMA is used; signature sequences may be correlated. When that happens, the graph over-estimates the actual ability to hold simultaneous wireless conversations in a network. Specifically, if two edges in the input graph are independent, this does not ensure that the SINR at the receiving node will be satisfactory. Examples of the shortcomings of the graph representation are depicted in Figures 3.1 and 3.2.

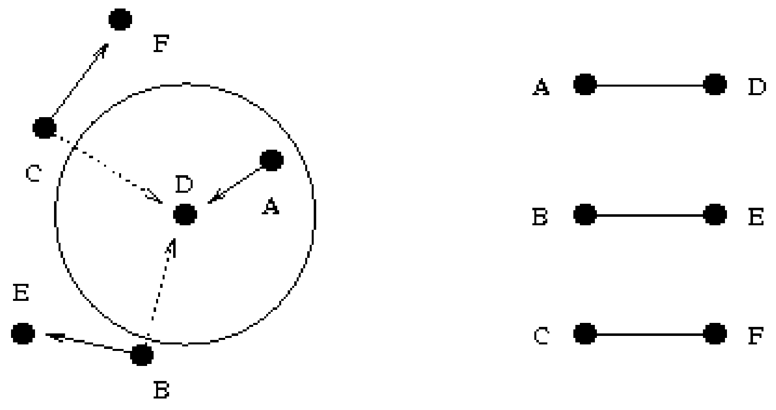


Figure 3.1: The matching $A \to D, B \to E, C \to F$. In the geometric picture at left, we see that the three links may be an infeasible matching, because D's SINR could be inadequate. (At left, solid lines represent intended transmissions, dotted lines represent interference, and the circle contains the nodes which can exceed D's SINR when interference is 0 (D's SNR).) This possibility is not apparent from the graph representation at right.

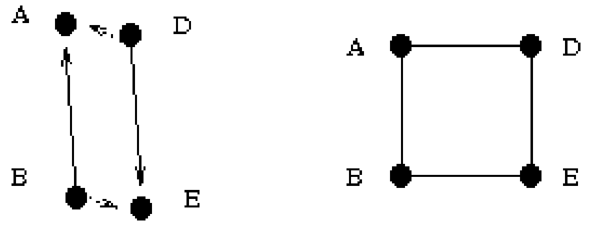


Figure 3.2: The matching $B \to A, D \to E$ is likely infeasible because A and E are overwhelmed by interference. Yet this possibility is not evident from the graph representation at right.

In this chapter, we examine the wireless minimum-length scheduling problem, with the constraints of [23], together with a constraint that has more often been associated with contention: an SINR condition. By accounting for interference, we are generalizing the model of Hajek and Sasaki [23], in which the only requirement was that no node can communicate with more than one neighbor at a time; there was no SINR requirement. (This case will appear in our model as the special case $\theta \approx 0$.) We assume the ability to control transmit power optimally, and for simplicity we allow arbitrarily large transmit powers.

Because of the constraint that a node cannot receive from more than one transmitter at a time, Hajek and Sasaki's results did not apply to networks whose receivers employ multiuser detection. Because we retain this constraint, our results do not apply there either. The effect of multiuser detectors would be to reduce schedule lengths.

With the SINR conditions, it is important that the link $i \to j$ be treated differently from $j \to i$, as transmissions originating at different nodes have differing interference effects. This is another important difference from [23], in which the graphs are undirected.

In the 1990s, much work on transmit power control was done, in the context of cellular networks. In those papers, concurrency is expressed as an SINR condition [2, 30, 29, 28]. More recently, it has been observed that transmit power control can lead to energy savings, an important advantage in many wireless systems. Some optimization along these lines can be found in [26], which uses a model similar to ours.

In this work we concentrate on the complexity of computing the minimum schedule length. We do not consider energy efficiency. We show in Section 3.4 that, in some cases, computing the minimum schedule length is tractable. In general it appears to be difficult to do so.

3.2 Notation and definitions

We assume a network of N wireless nodes. We are concerned with sets of k links where $2k \leq N$.

Definition 1. *A "matching" on the network is a set of simultaneous transmissions, $T_1 \rightarrow R_1, \ldots, T_k \rightarrow R_k$, with all $2k$ nodes distinct.*

It will sometimes be useful to think of a matching of k transmitters to k receivers as a one-to-one function from the set $\{1, \ldots, k\}$ onto itself, or a permutation π, with $T_i \rightarrow R_{\pi(i)}$.

Consider some matching. Let there be average noise power $\nu_j \geq 0$ at receiver j, $j = 1, \ldots, k$. Let $p_i \geq 0$ be the transmission power used by transmitter i, $i = 1, \ldots, k$. We denote the path loss from transmitter i to receiver j by α_{ij} where $0 < \alpha_{ij} \leq 1$.

Definition 2. *A matching $T_1 \rightarrow R_1, \ldots, T_k \rightarrow R_k$ is "feasible" if there exists a positive vector $p = (p_1, \ldots, p_k)^T$ of transmit powers such that the SINR condition*

$$\frac{p_i \alpha_{i,i}}{\nu_i + \sum_{m \neq i} p_m \alpha_{m,i}} > \theta \tag{3.1}$$

holds at each receiver R_i, $i = 1, \ldots, k$ in the matching, where the SINR threshold θ is assumed fixed throughout the network.

Recall that any nonzero square matrix with nonnegative elements has a largest eigenvalue ρ (called the Perron eigenvalue, Perron root or spectral radius) which is positive, and has a one-dimensional eigenspace containing an eigenvector with all positive components, called the Perron eigenvector [31, 32].

In the noiseless case, the problem of deciding whether a matching was feasible was shown in [2] to be an eigenvalue condition on a $k \times k$ nonnegative matrix.

Theorem 1 (Grandhi et al [2]). *Let L be the $k \times k$ matrix whose (i, j) element is α_{ij}. Consider the matching*

$$T_1 \to R_{\pi(1)}, T_2 \to R_{\pi(2)}, \ldots, T_k \to R_{\pi(k)}, \tag{3.2}$$

(a) the best SIR which can be achieved simultaneously by all k of the receivers in the matching is the reciprocal of the spectral radius of the $k \times k$ matrix

$$A(L, \pi) = \begin{bmatrix} 0 & \frac{\alpha_{2,\pi(1)}}{\alpha_{1,\pi(1)}} & \cdots & \frac{\alpha_{k,\pi(1)}}{\alpha_{1,\pi(1)}} \\ \frac{\alpha_{1,\pi(2)}}{\alpha_{2,\pi(2)}} & 0 & \cdots & \frac{\alpha_{k,\pi(2)}}{\alpha_{2,\pi(2)}} \\ \vdots & \vdots & & \vdots \\ \frac{\alpha_{1,\pi(k)}}{\alpha_{k,\pi(k)}} & \frac{\alpha_{2,\pi(k)}}{\alpha_{k,\pi(k)}} & \cdots & 0 \end{bmatrix} \tag{3.3}$$

(b) These SIRs are achieved by using the Perron eigenvector as a power vector.

We will refer to this matrix as $A(L, \pi)$, $A(\pi)$ or simply as A throughout this chapter and the next. Note that the (i, j) element of this matrix is

$$A(L, \pi)(i, j) = \begin{cases} 0 & i = j \\ \frac{L(j, \pi i)}{L(i, \pi i)} & i \neq j \end{cases} \tag{3.4}$$

In the noiseless case, any positive multiple of the Perron eigenvector, even a vanishingly small multiple, used as a power vector, will achieve SINR equal to $1/\rho(A)$ at all the receivers.

In this chapter and the next we will assume nonzero noises. When there is average noise power $(\nu_1, \ldots, \nu_k) \gneq (0, \ldots, 0)$ at the receivers, one can no longer achieve SINR equal to $1/\rho(A)$ but one can come arbitrarily close to this SINR value at each receiver by employing a sufficiently large multiple of the Perron eigenvector p.

Theorem 2 (Noisy version of Theorem 1). *Consider the matching in the previous theorem. Let average noise powers at the receivers be $(\nu_1, \ldots, \nu_k) \gneq (0, \ldots, 0)$. Let A be as in Equation (3.3) and let p be any Perron eigenvector of A. (a) For any $\epsilon > 0$ there exists a positive multiple of p such that SINR $\geq 1/\rho(A) - \epsilon$ at all k receivers. That is, $1/\rho(A)$ is the supremum of SINRs achievable at all receivers. (b) Given $\theta < 1/\rho(A)$, a sufficiently large positive multiple of p which achieves SINR$\geq \theta$ at all k receivers is $c \cdot p$ where*

$$c = \max_i \left\{ \frac{\nu_i/(\alpha_{i,\pi(i)}(\theta^{-1} - \rho(A)))}{p_i} \right\}. \tag{3.5}$$

Proof. (a) By using a sufficiently large positive multiple of p the noise can be made arbitrarily small in comparison to the interference (but not zero).

(b) Without loss of generality, suppose the matching $T_1 \to R_1, \ldots, T_k \to R_k$ is feasible. That is, given that $1/\rho(A) > \theta$, we can quickly compute a power vector $p = (p_1, \ldots, p_k)^T$ that achieves SINR $\geq \theta$ at all the receivers. It is any solution to the matrix inequality $((1/\theta)I - A)p > \eta$ where $\eta_i = \nu_i/\alpha_{ii}$. Let p equal any positive

Perron eigenvector of A, and ask what $c \cdot p$ we should use. By the definition of an eigenvector $((1/\theta)I - A)cp = (1/\theta - \rho(A))cp \geq \eta$. Thus for each $i = 1, \ldots, k$, we need $c \cdot p_i \geq \frac{\eta_i}{1/\theta - \rho(A)}$ or $c \geq \max_i \frac{\eta_i}{p_i(1/\theta - \rho(A))}$, as required. □

This theorem is the main reason for assuming unbounded transmit power values. The effect of limiting transmit power is unclear.

Definition 3. *A feasible matching is "maximal" if no link can be added to it without making it infeasible.*

Note that with this definition, there might be maximal matchings of several different sizes.

Among the N nodes there are E directed links which have positive demand which we must satisfy with our schedule, where $E \leq N(N-1)$. Define the "demand vector" $f = (f_1, \ldots, f_E)$ to have positive components f_ℓ equal to the time needed for the link ℓ to be active. This value f_ℓ can also be thought of as the amount of data to be transmitted over link ℓ in the schedule, since we assume the data rate is some constant value which can be supported by SINR $= \theta$.

A hypergraph (or set system) is a set V of vertices and a set S of hyperedges, where each hyperedge is a nonempty subset of the vertices. Associated with any hypergraph is an incidence matrix $Q = (q_{ij})$ where $q_{ij} = 1$ if vertex i is in hyperedge j and 0 otherwise.

3.3 The minimum length scheduling problem

We are given a set of N nodes, all the losses $0 < \alpha_{ij} \leq 1$, $i, j = 1, \ldots, N$ in the form of a square matrix L, the common SINR threshold $\theta > 0$ to be exceeded at a receiver for successful reception, the (possibly differing) non-negative average noise powers ν_i, $i = 1, \ldots, N$ at the receivers, and nonnegative demands f_{ij}, $i, j = 1, \ldots, N$ to be satisfied. It is assumed that $f_{ii} = 0$ for all nodes i.

A schedule is a set of matchings $M_1, \ldots, M_s$ and corresponding positive durations $\lambda_1, \ldots, \lambda_s$ such that each of the matchings is feasible, and the demands on all links are satisfied. If the sum of the durations is minimized, the schedule has minimum length. Applying the matchings in different orders certainly changes the schedule, but it will not affect the length of the schedule. Without loss of generality, we assume that every node x is an endpoint of some link of positive demand: $f_{ix} > 0$ or $f_{xj} > 0$ for some i or j. If not, node x does not need to be considered in scheduling and we could eliminate it from the problem.

Once we have found a set of feasible matchings and durations $\{\lambda_i, M_i\}_{i=1}^{s}$, we may then find the power vectors that each station should use by computing for each matching in the solution an eigenvector of the associated matrix $A(M_i)$, and choosing a sufficiently large multiple of it, as detailed above in Equation (3.5). We shall show below that it is always possible to find a solution consisting of $s \leq E$ matchings, so the work involved in finding transmit powers is no more than E eigenvector computations on matrices of size $N/2 \times N/2$. Alternately, the stations could determine the optimal power vector in a distributed manner as detailed in [28, 29].

We will be interested in the complexity of computing the minimum schedule length. Stated as a decision problem, an instance of "Minimum Length Scheduling" or MLS(L, θ, f, τ) is a positive rational number θ, an $N \times N$ matrix L of rationals in the interval $(0, 1]$, a nonnegative vector f of rationals, and a rational positive number τ. Question: Does there exist a finite set $M_1, \ldots, M_s$ of feasible matchings and associated positive durations $\lambda_1, \ldots, \lambda_s$ such that $\sum_{i=1}^{s} \lambda_i I(M_i) = f$ where $I(M)$ is the indicator vector of a matching and $\sum_{i=1}^{s} \lambda_i \leq \tau$?

3.3.1 LP formulation and hypergraphs

We construct a hypergraph $\mathcal{H}$ as follows. The vertices of the hypergraph are the links of positive demand, of which there are E. The hyperedges consist of all feasible matchings. We form the incidence matrix Q from this hypergraph $\mathcal{H}$.

Then the minimum schedule length is the value of

$$P: \quad \min \quad 1^T\lambda \tag{3.6}$$

$$\text{subj. to} \quad Q\lambda = f \tag{3.7}$$

$$\lambda \geq 0 \tag{3.8}$$

where λ is $W \times 1$, Q is $E \times W$ and f is $E \times 1$.

P is clearly a linear program. We can demonstrate that its value is bounded (so an optimal solution exists) as follows. Activate every link in a round robin fashion, one at a time, until its demand has been satisfied. This schedule has length

$\sum_i \sum_j f_{ij}$. An optimal schedule has length no greater than this value and no less than zero.

Since there exists an optimal solution, the theory of linear programming tells us that there must exist a basic optimal solution, i.e. one which is nonzero on no more than E of the λ's [33]. The existence of a basic optimal solution eliminates any fear that the problem might require an exponential amount of time to describe a solution.

However, the number W of feasible matchings (hyperedges) may be as large as 2^N. We might hope to avoid that problem by defining another hypergraph, $\mathcal{H}_m$, as follows. The vertices of $\mathcal{H}_m$ are the links of positive demand, as before. The hyperedges consist only of the W_m maximal feasible matchings.

Let Q_m be the incidence matrix associated with $\mathcal{H}_m$. Problem P_m defined by the linear program

$$P_m : \quad \min \quad 1^T \lambda \tag{3.9}$$

$$\text{subj. to} \quad Q_m \lambda \geq f \tag{3.10}$$

$$\lambda \geq 0 \tag{3.11}$$

has (by the same argument as above) a solution λ having no more than E nonzero components.

However the positive components of λ produced by P_m may exceed the demand f in some components ($Q_m \lambda > f$), which is an undesirable feature in a solution to the scheduling problem. Therefore we must say something about the ability to go back and forth between solutions of P and P_m.

The dual linear programs to P and P_m are

$$D: \quad \max \quad f^T u \tag{3.12}$$

$$\text{subj. to} \quad Q^T u \leq 1 \tag{3.13}$$

and

$$D_m: \quad \max \quad f^T u \tag{3.14}$$

$$\text{subj. to} \quad Q_m^T u \leq 1 \tag{3.15}$$

$$u \geq 0. \tag{3.16}$$

where Q_m^T is a $W_m \times E$ matrix whose columns are links of positive demand and whose rows are matchings.

Lemma 3. *Let the problems P, D, P_m, and D_m be defined by Equations (3.6), (3.12), (3.9), (3.14). Then*

1. *Solutions of D are solutions of D_m and vice versa.*
2. *All four problems have the same value, which is the minimum schedule length.*
3. *Solutions of P_m can be transformed into solutions for P in $\mathcal{O}(E^2)$ time.*
4. *Solutions of P can be transformed into solutions for P_m in $\mathcal{O}(EN^4)$ time.*

Proof. Let u be a solution of D_m. No u' having negative components could achieve $f^T u' > f^T u$ because $f > 0$. Therefore the nonnegativity of inequality (3.16) is unnecessary. The sets (3.13) and (3.15) of inequalities are the same, with the first set possibly containing redundant inequalities. Therefore the linear programs D and D_m have identical solution sets.

The second part of the Lemma follows from the strong duality of linear programming and the first part.

The proofs of the third and fourth parts are straightforward and are omitted.

$\square$

It is not clear how to transform solutions of duals into solutions of primals or vice versa. In ordinary linear programs complementary slackness would be used, but in this case applying complementary slackness appears to require exponential time due to the dimensions of the matrices Q and Q_m.

Q_m has fewer columns than Q, because every column of Q_m exists in Q but not the other way around. However, the example network of Figure 3.3 shows that even Q_m is still too large. Because Q and Q_m can be very wide, solving the linear programs P and P_m must be done in such a way that the matchings are never explicitly enumerated, for this would take time $\mathcal{O}(2^N)$.

In general the solutions of the dual D_m need not be 0-1 vectors. For example, if the dual is to maximize $u_1+u_2+u_3$ subject to $u_1+u_2 \leq 1$, $u_1+u_3 \leq 1$, $u_2+u_3 \leq 1$, $0 \leq u_1, u_2, u_3 \leq 1$, the optimal solution is $[1/2, 1/2, 1/2]$.

In the next section we consider a special case of the scheduling problem in which there is always a 0-1 solution for the dual problem, and for which there is a fast solution method.

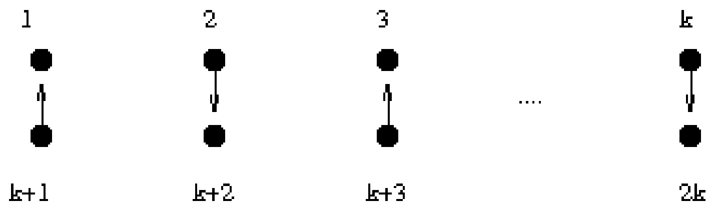


Figure 3.3: A network of $N = 2k$ nodes. Suppose that the losses and the SINR threshold are such that $i \rightarrow k+i$ or $k+i \rightarrow i$ can always be included in any matching, for $i = 1, \dots, k$, but inclusion of $i \rightarrow k+j$ or $k+j \rightarrow i$ make any matching infeasible for $j \neq i$. Then every matching in the above graph, where an up or down arrow connects every i to $k+i$, is a feasible maximal matching. There are $2^{N/2}$ maximal feasible matchings in this network. No matter how large θ is (large θ makes matchings less likely to be feasible) there exist losses α_{ij} reflecting this situation.

3.4 An algorithm to compute the min schedule length for superincreasing f

Definition 4. *A superincreasing vector a is one whose components, when sorted so that $a_1 \geq \cdots \geq a_n$, satisfy $a_i \geq \sum_{j>i} a_j$. We say a is strictly superincreasing if $a_i > \sum_{j>i} a_j$.*

Definition 5. *A submatching of a matching M is M with zero or more links removed. A supermatching of M is M with zero or more links added.*

Several facts, which will be useful in this chapter and the next, are proved in the following.

Lemma 4.

(a) Submatchings of a feasible matching are feasible.

(b) Supermatchings of an infeasible matching are infeasible.

(c) A single link $i \to j$ is a feasible matching if the maximum transmit power exceeds $\theta\nu_j/\alpha_{ij}$. (Here we assume all transmit powers are possible, so all single links are feasible matchings.)

(d) $i \to j$ and $k \to l$ coexist in some maximal matching $\Longleftrightarrow$ the matching $i \to j, k \to l$ is feasible.

Proof. (a) Let π be a matching and π' be a submatching of π. Then $A(\pi')$ is a principal submatrix of $A(\pi)$. The spectral radius of the principal submatrix cannot exceed the spectral radius of the matrix ([32] Cor. 8.1.20), and the result follows. Statement (b) is the contrapositive of (a). Statement (c) follows immediately from the SINR requirement (3.1). To prove (d), part (a) implies the forward direction ($\Longrightarrow$). The other direction is obvious. $\square$

Parts (a) and (b) of the lemma have a clear interpretation from an interference point of view. If a matching is feasible, then by removing some pairs the others have less interference to overcome, so the same power vector that worked in the matching will also work for the submatching. If a matching is infeasible, then the addition of other transmitter-receiver pairs will only add to the interference at the original receivers, which can only reduce their SINR for any power vector. Therefore an infeasible matching cannot be made feasible by adding links.

Theorem 5. *Let f be a superincreasing vector. Then there exists an optimal solution u to the dual problem D_m whose components are 0-1. Further, if f is strictly superincreasing, this 0-1 solution is the unique optimal solution.*

Proof. Order the components of u to correspond with the ordering of the components of f in the statement of the theorem.

We first show that the first component equals 1 in some optimal solution. Let u be an optimal solution to D_m and suppose $0 \leq u_1 < 1$ and that $0 \leq u_j \leq 1$ for $j > 1$. We will now construct a vector u' whose first component is 1 and whose other components are adjusted to maintain the feasibility of u', and show that the objective function value of u' is at least as great as that of u, which will imply u' is optimal.

The vector u has been assumed to solve D_m. Therefore it is feasible, i.e. it satisfies (3.15) and (3.16). Thus, for every maximal matching m (which is a column of Q_m) , we have $m^T u \leq 1$. The vector u' will be constructed by setting $u'_1 = 1$ and reducing components 2, 3, ..., E of u so that $m^T u' \leq 1$. Since only the first component of u' exceeds that of u, by $(1 - u_1)$, and m is a 0-1 vector, the difference between components $2, \ldots, E$ of u and u' can never be greater than $(1 - u_1)$:

$$u_j - u'_j \leq 1 - u_1 \tag{3.17}$$

for each $j > 1$.

The change in the objective function is

$$\begin{aligned} f^T u' - f^T u &= \left(f_1 + \sum_{j>1} u'_j f_j\right) - \left(u_1 f_1 + \sum_{j>1} u_j f_j\right) && (3.18)\\ &= (1 - u_1) f_1 - \sum_{j>1} f_j \left(u_j - u'_j\right) && (3.19)\\ &\geq (1 - u_1)\left(f_1 - \sum_{j>1} f_j\right) && (3.20)\\ &\geq 0 && (3.21) \end{aligned}$$

where the first inequality follows from Equation (3.17) and the second inequality from the superincreasing property of f. Since u was assumed optimal, and the new solution u' has at least as large an objective function value, u' is optimal.

Now we proceed by induction. Suppose that we know that there exists an optimal solution u whose first $r - 1$ components are 0 or 1. We will show that there exists another optimal solution u' whose first r components are 0-1.

If link r coexists in some feasible matching with some link among $1, \ldots, r - 1$

whose component in u is 1, then feasibility would require $u_r = 0$, and the proof would be complete. So suppose link r does not coexist in any feasible matching with any link among $1, \ldots, r-1$ whose component in u is 1, and suppose $0 < u_r < 1$. Let u' equal u in the first $r-1$ components and $u'_r = 1$. The difference in values of the objective functions is

$$\begin{aligned} f^T u' - f^T u &= \left(\sum_1^{r-1} u_i f_i + f_r + \sum_{j>r} u'_j f_j \right) - \left(\sum_1^{r-1} u_1 f_1 + u_r f_r + \sum_{j>1} u_j f_j \right) \\ &= (1-u_r) f_r - \sum_{j>r} f_j (u_j - u'_j) \qquad (3.22) \\ &\geq (1-u_r) \left(f_r - \sum_{j>r} f_j \right) \qquad (3.23) \\ &\geq 0 \qquad (3.24) \end{aligned}$$

because $u_j - u'_j \leq 1 - u_r$ for $j = r+1, \ldots, E$.

Since u was assumed optimal, u' must also be optimal, which completes the induction step, and proves that there exists a vector having 0-1 components which solves D_m.

If f is strictly superincreasing, then inequalities (3.21) and (3.24) become strict and the 0-1 solutions are strictly better than any non-0-1 competitors. This completes the proof of the theorem. $\square$

3.4.1 The algorithm for superincreasing f

We provide the following greedy algorithm, which solves problem D_m in $\mathcal{O}(N^4)$ operations when f is superincreasing. It constructs a 0-1 dual solution vector u explicitly.

First, precompute for every pair $i \to j, k \to l$ of links, with i, j, k, l distinct, whether this pair can coexist in a feasible matching. This can be done by computing the spectral radius ρ of the 2×2 matrix

$$A(i \to j, k \to l) = \begin{bmatrix} 0 & \frac{\alpha_{kj}}{\alpha_{ij}} \\ \frac{\alpha_{il}}{\alpha_{kl}} & 0 \end{bmatrix}. \tag{3.25}$$

If $\rho < 1/\theta$, the pair is feasible, so by part (d) of Lemma 4 the pair can coexist in some feasible matching.

Start with $u = (0, \ldots, 0)$. Go through the links in order of decreasing demand, and for each one decide whether u_ℓ will equal 0 or 1 as follows. If the current link ℓ can coexist with any link already assigned a 1 in this process, then u_ℓ is assigned 0. Otherwise u_ℓ is assigned 1.

Proof. Correctness of the algorithm: Since each component of u is either 0 or 1 at every step of the algorithm, inequality (3.16) is satisfied. No link ℓ is assigned $u_\ell = 1$ if it is involved in a matching with an already active link. By part (d) of Lemma 4 this satisfies constraint (3.15), so at all times u constitutes a feasible solution to D_m.

Now we show that the 0-1 solution obtained is optimal. The algorithm greedily adds the largest available component of f to its score without regard for later decisions. This is correct because of the superincreasing property. As we consider adding f_k to the score by setting $u_k = 1$, we stand to gain f_k on the score whereas the net improvement from all subsequent decisions will be less.

By Lemma 3, the optimal solution for the dual is the minimal schedule length. So this algorithm finds the minimum schedule length.

Running time of the algorithm: Creating the table of link compatibilities requires for each distinct i, j, k, l the computation of an eigenvalue of the 2×2 matrix of Equation (3.25) and a comparison to $1/\theta$, each of which has complexity $\mathcal{O}(1)$, so the precomputation costs $\mathcal{O}(N^4)$. Sorting the demands requires $\mathcal{O}(N^2 \log N)$. In greedily deciding whether to set a component of u to 1, we may have to consider N^2 link compatibilities. The list is N^2-long, so the greedy part requires $\mathcal{O}(N^4)$ operations. Thus the running time of this centralized algorithm is $\mathcal{O}(N^4)$. $\square$

It is interesting that although the algorithm computes the minimum schedule length, and explicitly computes a solution u to the dual problem, we still have not constructed any schedule which would achieve this length. This is the practical result of being unable to turn a dual solution into a primal solution.

Example We consider a 6-node network. The losses α_{ij} between all the nodes are (with rows and columns running from 1 to 6)

$$L = \begin{bmatrix} 1 & 0.0355 & 0.0042 & 0.0027 & 0.0016 & 0.0105 \\ 0.0303 & 1 & 0.004 & 0.0069 & 0.0028 & 0.0212 \\ 0.0036 & 0.0043 & 1 & 0.0066 & 0.0038 & 0.0011 \\ 0.0027 & 0.004 & 0.0071 & 1 & 0.0678 & 0.0018 \\ 0.0014 & 0.0031 & 0.0041 & 0.0614 & 1 & 0.0013 \\ 0.0097 & 0.0186 & 0.0015 & 0.0019 & 0.0014 & 1 \end{bmatrix}$$

We are given $\theta = 0.33$ and the demand vector $f = [1, 500, 17, 8, 37, 4, 2, 90]^T$ corresponding to links $1 \to 2$, $2 \to 3$, $3 \to 4$, $3 \to 6$, $4 \to 5$, $5 \to 3$, $5 \to 6$, and $6 \to 1$ respectively.

We begin with $u = (0,0,0,0,0,0,0,0)^T$. We immediately set $u_{2\to3} = 1$ because link $2 \to 3$ corresponds to the largest component of f.

We consider $u_{6\to1}$. Since the spectral radius of

$$A(2 \to 3, 6 \to 1) = \begin{bmatrix} 0 & \frac{\alpha_{63}}{\alpha_{23}} \\ \frac{\alpha_{21}}{\alpha_{61}} & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0.375 \\ 3.12 & 0 \end{bmatrix} \tag{3.26}$$

equals $\rho = 1.08$, which is less than $1/.33$, the two links are feasible together so we must set $u_{6\to1} = 0$.

We next consider $u_{4\to5}$. By another 2×2 eigenvalue computation we find that the matching $4 \to 5$, $2 \to 3$ is feasible so $u_{4\to5} = 0$.

Consider $u_{3\to4}$. The link $3 \to 4$ is not a matching with $2 \to 3$ because node 3 cannot simultaneously transmit and receive, so set $u_{3\to4} = 1$.

Consider $u_{3\to6}$. The link $3 \to 6$ cannot be part of a feasible matching with either $2 \to 3$ or $3 \to 4$ because all involve node 3, so set $u_{3\to6} = 1$.

Similarly we set $u_{5\to3} = 1$.

To decide on $u_{5\to6}$, we note that $5 \to 6$ could not be part of a feasible matching with either $3 \to 6$ or $5 \to 3$. We compute $\rho(A(5 \to 6, 2 \to 3)) = 4.08$ so the matching $5 \to 6$,$2 \to 3$ is not feasible. Finally $\rho(A(5 \to 6, 3 \to 4)) = 2.81$ so the matching $5 \to 6$,$3 \to 4$ is feasible. Consequently $u_{5\to6} = 0$.

Finally, we consider the link corresponding to the smallest positive demand, $1 \to 2$. We compute $\rho(A(1 \to 2, 3 \to 4)) = 0.22$, so set $u_{1\to2} = 0$.

We have computed $u = (0,1,1,1,0,1,0,0)^T$, which is guaranteed to be an optimal solution for the dual. This solution has value $f^T u = 529$. Therefore there must exist a schedule of this length. Through a direct computation, which can be completed in a reasonable time only because this instance of the problem is small, we can find the following schedule:

feasible matching	duration
$2 \to 3,\ 4 \to 5,\ 6 \to 1$	37
$2 \to 3,\ 6 \to 1$	53
$2 \to 3$	410
$3 \to 4,\ 5 \to 6$	2
$1 \to 2,\ 3 \to 4$	1
$3 \to 4$	14
$3 \to 6$	8
$5 \to 3$	4
	529

3.4.2 Other tractable sub-cases

We just showed that MLS(L, θ, f, τ) is solvable in polynomial time when f is super-increasing. We note a few other tractable sub-cases of MLS.

There exists a positive number $\theta_{safe}(*)$ such that if $0 \leq \theta < \theta_{safe}(*)$ then all matchings are feasible, i.e. we can disregard the SINR constraint (3.1), because it is automatically satisfied. (Chapter 4 examines θ_{safe}.) If our input θ is less than $\theta_{safe}(*)$ we can form an undirected weighted graph, with an edge (i, j) between nodes i and j if $i \to j$ or $j \to i$ has positive demand, and to this edge assign weight $f_{i \to j} + f_{j \to i}$. Then the polynomial time algorithm of Hajek and Sasaki [23] will determine the shortest schedule. This is a polynomial time algorithm, then, for

MLS when θ is small. For this reason our model can be seen as a generalization of the model from [23].

The special case of MLS where losses between all nodes are equal and all the demands are equal is also solvable in polynomial time. One determines the size of the largest feasible matching. By symmetry, all matchings of this size must be feasible, and a schedule of minimum length can be made by applying all these matchings for an equal time. The minimum schedule length can then be computed easily.

If MLS is modified so that we restrict the size of matchings to K or fewer links, then the LP no longer has exponential size and the problem can be solved in polynomial time.

3.5 Complexity of the general SINR-constrained scheduling problem

As indicated in Theorem 3, solving MLS is equivalent to solving the linear program D. D may have an exponential number of constraints. As a consequence of the existence of the ellipsoid algorithm for linear programming [34], D has a polynomial-time solution if and only if the "separation problem" has a polynomial-time solution. We now consider the separation problem.

The separation problem is, for any candidate vector u, to either (a) conclude that u is in the feasible region $Q^T u \leq 1$, or (b) find a violated inequality q satisfying $q^T u > 1$. The matrix Q only has 0-1 entries. Therefore a violated inequality is some feasible matching containing edges whose corresponding components of u sum to more than 1.

Let a network of $2k$ nodes be given, with k even, labeled $T_1, \ldots, T_k, R_1, \ldots, R_k$ and assume that there are only k^2 links of positive demand, all of the form $T_i \rightarrow R_j$ for $i, j = 1, \ldots, k$. The loss matrix L' for the network looks like $\begin{pmatrix} * & L \\ & \\ * & * \end{pmatrix}$ where the asterisks represent $k \times k$ submatrices whose values are unimportant. We must be able to solve the separation problem for all u; in particular we must be able to solve it for the $k^2 \times 1$ vector $u = (1/k + \epsilon, \ldots, 1/k + \epsilon)^T$ where $0 < \epsilon < 1/(k(k-1))$. Any k of the components of u sum to a number greater than 1, while any $k-1$ components sum to less than 1. So the separation problem becomes, "Does there exist no feasible matching of k links, and if there is one, exhibit it."

We now formalize this observation. Define the decision problem MAX-SINR-MATCHING(L,T) whose input is a rational $k \times k$ matrix L and a real threshold T as follows. Interpret $L(i, j)$ as the loss from transmitter i to receiver j, in some network of $2k$ nodes $T_1, \ldots, T_k$, $R_1, \ldots, R_k$. The question is, "is there a 1-1 mapping $T_1 \rightarrow R_{\pi(1)}, T_2 \rightarrow R_{\pi(2)}, \ldots, T_k \rightarrow R_{\pi(k)}$ of transmitters to receivers, such that for some positive vector $(p_1, p_2, \ldots, p_k)^T$ of transmit powers, the SINRs at all k of the receivers exceed T?"

We can show that MAX-SINR-MATCHING is in the class NP. For given the 1-1 mapping π, we form the matrix $A(\pi)$, compute the maximum achievable SINR and compare it to T. This can be done in polynomial time.

We know that answering "yes" to MAX-SINR-MATCHING is equivalent to the matrix $A(L, \pi)$ having its spectral radius less than $1/T$, for some permutation π. If there is a polynomial time algorithm to decide MAX-SINR-MATCHING, this is not enough to imply that MLS is easy, because we have not solved the separation problem for all possible candidates u in the dual problem. However, if there is no polynomial time algorithm for MAX-SINR-MATCHING we can conclude that the separation problem is not solvable in polynomial time.

Thus we have shown

Theorem 6. *If $P \neq NP$, and there is no polynomial time algorithm for MAX-SINR-MATCHING(L,θ) then there is no polynomial time algorithm for MLS(L, θ, f,τ).*

There are reasons to believe that MAX-SINR-MATCHING may be a hard problem. First, the obvious algorithm for solving it would be to compute $k!$ matrices $A(\pi)$ and for each one compute the Perron eigenvalue. The reciprocal of the smallest Perron eigenvalue would be the maximum achievable SINR.

Our intuition about what matchings would work best is as follows. Consider the $2k$ nodes in a plane. Since all $2k$ nodes must be active, what we want is for each transmitter to transmit to a nearby receiver. Then its transmit power can be low and it will interfere little with other receivers, as in Figure 3.3. In terms of the $k \times k$ loss matrix, we can choose exactly one loss from each row and each column, which will define a permutation of transmitters to receivers. If we are looking for the permutation yielding the best SINR, our intuition says that these k chosen entries should be relatively large, because harmful interference flows to the intended receivers on the losses we do not choose.

In the $k = 2$ case the max achievable SINR is $\sqrt{\alpha_{1,\pi(1)}\alpha_{2,\pi(2)}/\alpha_{1,\pi(2)}\alpha_{2,\pi(1)}}$, which may lead us to believe, for $k > 2$, that the permutation π corresponding to a large product $\alpha_{1,\pi(1)}\alpha_{2,\pi(2)} \cdots \alpha_{k,\pi(k)}$ of losses, one from each row and each column, would be a good choice. However the following counterexample shows that the permutation having largest product of losses is not always the permutation having the best SINR. The matrix

$$L_1 = \begin{bmatrix} .148 & .03 & .23 \\ .44 & .33 & .68 \\ .412 & .64 & .09 \end{bmatrix} \tag{3.27}$$

has its smallest spectral radius from the matching $T_1 \to R_1$, $T_2 \to R_3$, $T_3 \to R_2$. On the other hand the largest product of three elements of L, one from each row and column, is $(.23)(.44)(.64)$ which suggests the matching $T_1 \to R_3$, $T_2 \to R_1$, $T_3 \to R_2$.

Similarly one can supply examples to show that the permutation corresponding to the largest sum of losses need not have the best SINR.

One might expect that if there is some loss $L(i,j)$ such that $L(i,j) > L(i',j)$ for all $i' \neq i$ and $L(i,j) > L(i,j')$ for all $j' \neq j$ then transmitter i ought to be paired with receiver j. However the following counterexample shows that this is not necessarily the case.

$$L_2 = \begin{bmatrix} .407 & .187 & .404 \\ .339 & .315 & .346 \\ .064 & .542 & .394 \end{bmatrix} \tag{3.28}$$

The $(1,1)$ entry is both a row max and a column max, suggesting $T_1 \to R_1$ should be part of the optimal matching, but the pairing of maximum SINR is actually $T_1 \to R_3, T_2 \to R_1, T_3 \to R_2$.

This preliminary evidence suggests MAX-SINR-MATCHING may be a difficult problem. If it is, the Minimum Length Scheduling problem is also difficult, by Theorem 6.

3.6 A theorem on MAX-SINR-MATCHING

In this section we prove a theorem which expands the understanding of the MAX-SINR-MATCHING(L,T) problem. The theorem allows us to impose structure on the L matrix, such as assuming that L is doubly stochastic, without losing generality.

Define

$$\mathscr{G}(L,\pi) = I + A(L,\pi), \tag{3.29}$$

where A is the $k \times k$ matrix defined in (3.3) and (3.4). The addition of the identity matrix shifts the spectrum by 1, so

$$\rho(\mathscr{G}(L,\pi)) = 1 + \rho(A(L,\pi)). \tag{3.30}$$

It follows that π minimizes the spectral radius of A if and only if it minimizes the spectral radius of $\mathscr{G}$. In this section we prefer to examine $\mathscr{G}$ because its (i,j) element can compactly be described as

$$\mathscr{G}(L,\pi)(i,j) = \frac{L(j,\pi i)}{L(i,\pi i)}, \tag{3.31}$$

an improvement over Equation (3.4).

Definition 6. *A "generalized permutation matrix" is a square matrix whose zero pattern is that of a permutation matrix, i.e. if all its nonzero entries were changed to 1 it would be a permutation matrix.*

Any nonnegative generalized permutation matrix of size $k \times k$ has exactly k positive entries. It can be written as the product of a diagonal matrix with positive entries on the diagonal and a permutation matrix.

To avoid a more cumbersome notation, we write πi in place of $\pi(i)$. Also, the notation $\pi_1 \circ \pi_2$ means the composition of permutations as functions, so $(\pi_1 \circ \pi_2)i$ represents $\pi_1(\pi_2(i))$.

Theorem 7. *Let $\mathscr{G}(L, \pi)$ be the matrix defined in Equation (3.31). Let L be a $k \times k$ loss matrix, and π, q, p be permutations of the set $\{1, \ldots, k\}$. Let D, D_1 and D_2 be diagonal real matrices whose diagonal elements are positive. Let Q and P be the permutation matrices corresponding to q and p, respectively, in the sense that if $v = [1, 2, \ldots, k]^T$ then $Pv = [\ p(1), p(2), \ldots, p(k)]^T$. Then*

1. $\mathscr{G}(LD, \pi) = \mathscr{G}(L, \pi)$.
2. $\mathscr{G}(DL, \pi) = D^{-1}\mathscr{G}(L, \pi)D$.
3. $\mathscr{G}(D_1LD_2, \pi) = D_1^{-1}\mathscr{G}(L, \pi)D_1$.
4. $\mathscr{G}(PL, \pi) = P\mathscr{G}(L, \pi \circ p^{-1})P^{-1}$.
5. $\mathscr{G}(LQ, \pi) = \mathscr{G}(L, q^{-1} \circ \pi)$.
6. $\mathscr{G}(PLQ, \pi) = P\mathscr{G}(L, q^{-1} \circ \pi \circ p^{-1})P^{-1}$.

7. *$\mathscr{G}(PLP^{-1},\pi) = P\mathscr{G}(L, p\circ\pi\circ p^{-1})P^{-1}$.*

8. *If V is a lower triangular matrix then the two matrices $\mathscr{G}(LV,\pi)$ and $\mathscr{G}(L,\pi)$ are equal on row $\pi^{-1}(k)$.*

9. *Let $G_1 = PD_1$, $G_2 = D_2Q$ be nonnegative generalized permutation matrices. Then $\mathscr{G}(G_1LG_2,\pi) = PD_1^{-1}\mathscr{G}(L, q^{-1}\circ\pi\circ p^{-1})D_1P^{-1}$.*

Proof. (1) The (i,j) element of LD is $LD(i,j) = L(i,j)D(j,j)$. Using Equation (3.31), the (i,j) element of $\mathscr{G}(LD,\pi)$ is

$$\frac{LD(j,\pi i)}{LD(i,\pi i)} = \frac{L(j,\pi i)D(\pi i,\pi i)}{L(i,\pi i)D(\pi i,\pi i)} \tag{3.32}$$

which is the (i,j) element of $\mathscr{G}(L,\pi)$.

(2) The (i,j) element of DL is $LD(i,j) = D(i,i)L(i,j)$. Using Equation (3.31) the (i,j) element of $\mathscr{G}(DL,\pi)$ is

$$\frac{DL(j,\pi i)}{DL(i,\pi i)} = \frac{D(j,j)}{D(i,i)}\frac{L(j,\pi i)}{L(i,\pi i)} \tag{3.33}$$

which the the (i,j) element of $D^{-1}\mathscr{G}(L,\pi)D$.

Part (3) follows from parts (1) and (2).

(4) The (i,j) element of PL is $PL(i,j) = L(pi,j)$. Using Equation (3.31), the (i,j) element of $\mathscr{G}(PL,\pi)$ is

$$\frac{PL(j,\pi i)}{PL(i,\pi i)} = \frac{L(pj,\pi i)}{L(pi,\pi i)} \tag{3.34}$$

which equals the (pi,pj) element of $\mathscr{G}(L,\pi\circ p^{-1})$. This equals, by straightforward manipulation, the (i,j) element of $P\mathscr{G}(L,\pi\circ p^{-1})P^{-1}$.

(5) The (i,j) element of LQ is $LQ(i,j) = L(i, q^{-1}j)$. Using Equation (3.31), the (i,j) element of $\mathscr{G}(LQ,\pi)$ is

$$\frac{LQ(j,\pi i)}{LQ(i,\pi i)} = \frac{L(j,(q^{-1}\circ\pi)i)}{L(i,(q^{-1}\circ\pi)i)} \tag{3.35}$$

which equals the (i,j) element of $\mathscr{G}(L, q^{-1}\circ\pi)$.

Part (6) follows from parts (4) and (5).

To prove (7), let $Q = P^{-1}$ in part (6).

(8) The $(\pi^{-1}k, j)$ element of $\mathscr{G}(LV,\pi)$ is

$$\frac{\sum_m L(j,m)V(m,k)}{\sum_m L(\pi^{-1}k,m)V(m,k)} = \frac{L(j,k)}{L(\pi^{-1}k,k)} \tag{3.36}$$

because $V(m,k)$ is nonzero only when $m = k$. Using Equation (3.31) we can verify that the $(\pi^{-1}k, j)$ element of $\mathscr{G}(L,\pi)$ equals the same quotient.

Part (9) follows from parts (3) and (6). This completes the proof of Theorem 7.

$\square$

The collection of relations in Theorem 7 allows us to restate the MAX-SINR-MATCHING problem in equivalent forms.

Corollary 8. *1. Without loss of generality we may assume all loss matrices L are column stochastic.*

2. Without loss of generality we may assume all loss matrices L are doubly stochastic.

3. Re-numbering the nodes does not affect the maximum achievable SINR in the network, $1/\rho(A)$, but permutes the power vector which achieves the best SINR.

Proof. Part 1 follows from Part 1 of Theorem 7 by letting the diagonal elements of D be the inverses of the column sums of L.

By a theorem of Sinkhorn [35], every strictly positive matrix X can be transformed into a unique strictly positive doubly stochastic matrix S by $S = D_1 X D_2$ where D_1 and D_2 are diagonal with positive entries on the diagonal. Since similarity transforms such as $D^{-1}\mathscr{G}D$ leave the spectrum unchanged, and L is strictly positive, part 3 of the theorem shows we lose no generality in assuming L is doubly stochastic, which proves part 2 of the Corollary.

Part 7 of the theorem shows the effect of re-numbering the nodes. When nodes are renumbered by some permutation q the loss matrix becomes QLQ^{-1} and the A-matrix changes in such a way that the spectrum is unaffected, so the max SINR is not affected. The power vector required to achieve the max SINR is permuted accordingly. $\square$

3.7 Finding maximal matchings

In performing scheduling, it is of interest to be able to produce matchings in which many links are simultaneously active. It is also of interest to ask for matchings which contain as many links as possible in addition to some specified set. These matchings offer parallelism and will tend to produce short schedules.

It is desirable to be able to say, without exhaustive enumeration, what size feasible matchings are likely to exist. Given a loss matrix L for a network, and an SINR threshold θ, can we

1. predict the size of the maximal matching(s) without partitioning the nodes into sets of k transmitters and receivers, and for every k-matching, performing an eigenvalue computation?

2. find any single feasible matching of largest absolute size in a network?

3. find the matching involving all the transmitters from a set $\mathcal{T}$ and all the receivers from a set $\mathcal{R}$ whose A-matrix has smallest spectral radius?

All these problems appear hard. The set of all maximal matchings is not a matroid. The third problem is MAX-SINR-MATCHING, for which we have already given some preliminary evidence of difficulty. In the next section we present a sub-optimal heuristic for solving MAX-SINR-MATCHING.

3.7.1 A heuristic for MAX-SINR-MATCHING

A heuristic which runs in polynomial time is described here. Given L, we first make it doubly stochastic, to make all the losses comparable, using the iterative method of Sinkhorn [35]. Briefly, this method first makes its input column stochastic, then row stochastic, then column stochastic, and so forth. Performing these iterations takes the input from L to $D_1 L D_2$ where the column normalizations determine D_2 and the row normalizations determine D_1. As we saw in Corollary 8 this does not affect the resulting $\rho(A(L, \pi))$ for any π.

We then choose k entries of $D_1 L D_2$, one from each row and each column. The choice is made to maximize the product of these entries. Intuitively, this chooses generally large channel gains for intended transmissions, and generally smaller ones for interfering pairs. The optimal choice can be made in $\mathcal{O}(k^3)$ operations by using the assignment algorithm. The assignment algorithm [36] finds an assignment (k entries of the matrix, exactly one in each row and column of L) whose sum is maximized. We adapt the assignment algorithm by feeding it the matrix whose entries are logarithms of the entries of L so that it will return the assignment of maximum product.

The total time to run the algorithm is the time to run the iterative method of Sinkhorn, which takes $\mathcal{O}(k^2)$ per iteration, plus the $\mathcal{O}(k^3)$ time for the assignment algorithm. In practice, making D_1LD_2 close to doubly stochastic is good enough, because we are doing this to make all the losses in D_1LD_2 comparable. So if we allow ak Sinkhorn iterations for some fixed a, the whole algorithm will require $\mathcal{O}(k^3)$ operations.

As we saw in (3.27), the maximum-product heuristic is not perfect. The heuristic works well for smaller matrices. The following experiment was performed to evaluate the effectiveness of the maximum product heuristic.

A network of $2k$ nodes was placed randomly in a square area with the property that the worst case propagation loss between nodes was approximately 30 dB. Of the $2k$ nodes, half were chosen randomly as transmitters and half as receivers, to yield a $k \times k$ matrix L of losses. The MAX-SINR-MATCHING π was found using exhaustion, and the choice of the maximum product heuristic, π_h, was also found.

We measured the average ratio $\rho(A(\pi_h))/\rho(A(\pi))$ and the fraction of time that $\pi = \pi_h$. The following success rates were observed. For each value of k, 1000 or more trials were run.

Table 3.1: Experimental results for the “maximum product” heuristic.

size of L submatrix	observed $Pr(\pi = \pi_h)$	mean ratio $\frac{\rho(A(\pi_h))}{\rho(A(\pi))}$
3×3	99.7%	1.00003
4×4	98.3%	1.00019
5×5	96.1%	1.00051
6×6	92.9%	1.00128
7×7	91.8%	1.00136
8×8	88.8%	1.00223

3.8 Conclusion

We have considered the problem of scheduling to satisfy given demands in a wireless network which is constrained by SINR requirements. We showed that this problem can be seen as a generalization of the scheduling problem considered by Hajek and Sasaki in [23] for which a polynomial time algorithm was available.

In the special case where the demand vector has a superincreasing property we provide a greedy algorithm which efficiently computes the smallest schedule length. Although we can determine the length of the shortest schedule, we were unable to construct a schedule which realizes it.

In the general case, we showed that the minimum length scheduling problem with SINR constraints is at least as hard as the MAX-SINR-MATCHING problem, which appears difficult.

We presented a heuristic for the MAX-SINR-MATCHING problem. The heuristic chooses the matching whose product of channel gains is largest. This heuristic can be made to run in time $\mathcal{O}(k^3)$, where the input matrix of losses has dimensions $k \times k$. The heuristic performs well in experiments.

Chapter 4

The feasibility of matchings in a wireless network

4.1 Introduction

In this chapter we continue to exploit the connection of the SINR model of wireless networks to the rich theory of eigenvalues of nonnegative matrices [31, 40]. We prove theorems that show how to infer whether certain sets of links are feasible or not, without actually computing eigenvalues. Some of the theorems in this chapter allow us to make conclusions based only on the weakest link in the network.

Except where otherwise stated, the network model and notation are the same as in the previous chapter.

4.2 Results on feasibility of matchings

Our main result is the following:

Theorem 9. *Let the SINR threshold $\theta > 1$. Suppose some matching of k transmitters to k receivers is feasible. Then no permutation of the receivers among the transmitters, except the identity, yields a feasible matching.*

Proof. Proof is by induction on k. First we will prove the result for $k = 2$. Let $T_1 \to R_1, T_2 \to R_2$ be a feasible matching. Its matrix

$$A = \begin{bmatrix} 0 & \frac{\alpha_{21}}{\alpha_{11}} \\ \frac{\alpha_{12}}{\alpha_{22}} & 0 \end{bmatrix} \tag{4.1}$$

has (by straightforward calculation) eigenvalues $\pm\sqrt{\frac{\alpha_{12}\alpha_{21}}{\alpha_{11}\alpha_{22}}}$. Now consider the matching $T_1 \to R_2, T_2 \to R_1$ which is the first matching with the receivers permuted. Its matrix

$$B = \begin{bmatrix} 0 & \frac{\alpha_{22}}{\alpha_{12}} \\ \frac{\alpha_{11}}{\alpha_{21}} & 0 \end{bmatrix} \tag{4.2}$$

happens to be the inverse of A so its eigenvalues are the reciprocals of those of A, $\pm\sqrt{\frac{\alpha_{11}\alpha_{22}}{\alpha_{12}\alpha_{21}}}$. By inspection we can see that the largest eigenvalues of these two matrices are reciprocals, so both cannot be smaller than $1/\theta$ where $\theta > 1$. Therefore by Theorem 2, no more than one of the two systems can be feasible. Thus the result is true for $k = 2$.

Suppose the result is true for all numbers of pairs less than k, and consider a network of k transmitter-receiver pairs for which we have a feasible matching, say $T_1 \to R_1$, $T_2 \to R_2$, …, $T_k \to R_k$. Suppose we permute the receivers with respect to the transmitters by some non-identity permutation π which is the product of two or more disjoint cycles, i.e. $\pi = \sigma_1\sigma_2\cdots\sigma_r$, $r \geq 2$. Since π is not the identity, the length of one of these cycles is $2 \leq \ell \leq k-1$. Without loss of generality let

σ_1 be the cycle $(1, 2, \ldots, \ell)$. By part (a) of Lemma 4, the submatching $T_1 \to R_1$, $T_2 \to R_2$, …, $T_\ell \to R_\ell$ is feasible. After the application of σ_1, $T_1 \to R_2$, $T_2 \to R_3$, …, $T_{\ell-1} \to R_\ell$, $T_\ell \to R_1$ is infeasible, by the induction hypothesis. By part (b) of the Lemma, the supermatching π of σ_1 is also infeasible. Therefore the Theorem is true when π is the product of two or more disjoint cycles.

Suppose that π is a single cycle. To obtain a contradiction suppose that among the k transmitters and k receivers, both of the matchings

$$T_1 \to R_1, T_2 \to R_2, \quad \ldots, T_k \to R_k$$

$$T_1 \to R_{\pi(1)}, T_2 \to R_{\pi(2)}, \quad \ldots, T_k \to R_{\pi(k)}$$

are feasible, i.e. that some k-dimensional power vector (not necessarily the same one for each matching) exceeds the SINR requirements of the receivers.

By Theorem 2, the following matrices A and B, which correspond to the first and second matchings respectively, must each have their Perron eigenvalue less than $1/\theta$.

$$A = \begin{bmatrix} 0 & \frac{\alpha_{21}}{\alpha_{11}} & \cdots & \frac{\alpha_{k1}}{\alpha_{11}} \\ \frac{\alpha_{12}}{\alpha_{22}} & 0 & \cdots & \frac{\alpha_{k2}}{\alpha_{22}} \\ & \vdots & & \\ \frac{\alpha_{1k}}{\alpha_{kk}} & \frac{\alpha_{2k}}{\alpha_{kk}} & \cdots & 0 \end{bmatrix} \tag{4.3}$$

$$B = \begin{bmatrix} 0 & \frac{\alpha_{2,\pi(1)}}{\alpha_{1,\pi(1)}} & \cdots & \frac{\alpha_{k,\pi(1)}}{\alpha_{1,\pi(1)}} \\ \frac{\alpha_{1,\pi(2)}}{\alpha_{2,\pi(2)}} & 0 & \cdots & \frac{\alpha_{k,\pi(2)}}{\alpha_{2,\pi(2)}} \\ & \vdots & & \\ \frac{\alpha_{1,\pi(k)}}{\alpha_{k,\pi(k)}} & \frac{\alpha_{2,\pi(k)}}{\alpha_{k,\pi(k)}} & \cdots & 0 \end{bmatrix} \tag{4.4}$$

Now consider the $k \times k$ matrix $\widetilde{B}$ obtained by reducing to zero most of the entries of B, defined by

$$\widetilde{B}_{ij} = \begin{cases} 0 & if\ \pi(i) \neq j \\ \alpha_{jj}/\alpha_{ij} & if\ \pi(i) = j \end{cases} \tag{4.5}$$

Because π is one-to-one, $\widetilde{B}$ will have no more than one nonzero element in each row and column. Because π has no fixed points, $\widetilde{B}$ will have exactly one nonzero element in each row and column. Similarly define $\widetilde{A}$ by setting most of A's entries to zero:

$$\widetilde{A}_{ij} = \begin{cases} 0 & if\ i \neq \pi(j) \\ \alpha_{ji}/\alpha_{ii} & if\ i = \pi(j) \end{cases} \tag{4.6}$$

Note that $0 \leq \widetilde{A} \leq A$ and $0 \leq \widetilde{B} \leq B$ entrywise.

Consider the product matrix $\widetilde{A}\widetilde{B}$. Its (i, j) entry is

$$\sum_{m=1}^{k} \widetilde{A}_{im}\widetilde{B}_{mj} = \left(\delta_{i,\pi(m)}\frac{\alpha_{mi}}{\alpha_{ii}}\right)\left(\delta_{\pi(m),j}\frac{\alpha_{jj}}{\alpha_{mj}}\right) = \delta_{ij} \tag{4.7}$$

Evidently $\widetilde{A}$ and $\widetilde{B}$ are inverses. From the fact that $\widetilde{A}$ is a generalized permutation matrix, its k distinct eigenvalues are β times the k roots of unity where β is the positive constant $\left(\prod_{m=1}^{k} \frac{\alpha_{\pi^{-1}(m),m}}{\alpha_{m,m}}\right)^{1/k}$. Therefore $\widetilde{A}$ has only one real positive eigenvalue, β. Since $\widetilde{B}$ is $\widetilde{A}$'s inverse its k distinct eigenvalues must lie on the circle of radius $1/\beta$ centered at the origin, and $\widetilde{B}$ also has only one positive eigenvalue, $1/\beta$.

We have determined that the Perron eigenvalues of $\widetilde{A}$ and $\widetilde{B}$ are β and $1/\beta$. Only one of these could be smaller than $1/\theta$. Since $A \geq \widetilde{A} \geq 0$ and $B \geq \widetilde{B} \geq 0$, the Perron eigenvalues of A and B are at least as large as those of $\widetilde{A}$ and $\widetilde{B}$. Therefore it is impossible that both $\rho(A)$ and $\rho(B)$ are both less than $1/\theta$, which contradicts the assumption that both matchings are feasible.

Thus the claim is true for k and the proof by induction is complete. $\square$

When considering k transmitters and k receivers, there are $k!$ ways to assign the receivers to the transmitters. The theorem says that no more than one of those matchings can be feasible, if the SINR threshold $\theta > 1$.

4.3 Where all matchings are feasible

Theorem 9 shows that there is not much flexibility in network design when $\theta > 1$. However θ may be less than 1. For example CDMA may be thought of as effectively reducing θ by a factor equal to the spreading gain. Therefore it is desirable to say something about what permutations of the receiver set are feasible when $\theta < 1$.

Suppose we want to design a network in which all matchings are feasible, so that we can freely reassign receivers and transmitters while remaining sure that feasible power vectors will exist. (Given existence of the power vector, a distributed algorithm for converging to it exists and was derived in [28].)

As we saw in Section 3.4.2, minimum length scheduling becomes tractable when we can ignore the SINR constraints.

Definition 7. *Let two disjoint sets $\mathcal{T} = \{T_1, \ldots, T_k\}$ and $\mathcal{R} = \{R_1, \ldots R_k\}$ of transmitters and receivers be given. We define a critical value $\theta_{safe}(\mathcal{T}, \mathcal{R})$ to be the supremum of the SINR thresholds under which all $k!$ matchings of the receivers in $\mathcal{R}$ to the transmitters in $\mathcal{T}$ are feasible.*

Definition 8. *Let $\theta_{safe}(k)$ be the largest value of θ such that every matching of k links in the network is feasible.*

Definition 9. *Let $\theta_{safe}(*)$ be the largest value of θ such that every matching in the network is feasible.*

Clearly, for each of the above definitions, $\theta_{safe} \geq 0$ since any matching will be feasible if the SINR requirement is zero. Theorem 9 showed that $\theta_{safe} \leq 1$, since no more than one matching could be feasible when $\theta > 1$.

We have the following relationship:

$$\theta_{safe}(*) \leq \theta_{safe}(k) \leq \theta_{safe}(\mathcal{T}, \mathcal{R}) \tag{4.8}$$

if $|\mathcal{T}| = |\mathcal{R}| \leq k \leq N/2$.

Theorem 10. *If the $k \times k$ matrix L has (i, j)th element the loss from transmitter $i \in \mathcal{T}$ to receiver $j \in \mathcal{R}$ then*

$$\theta_{safe}(\mathcal{T}, \mathcal{R}) = [\max_{\pi} \rho(A(L, \pi))]^{-1} \tag{4.9}$$

Proof. This follows from Theorems 1 and 2. Since θ_{safe} is defined as a supremum, it doesn't matter whether we have noise or not. $\square$

The following theorems give lower and upper bounds on $\theta_{safe}(\mathcal{T}, \mathcal{R})$.

Proposition 11 (upper bound on $\theta_{safe}(\mathcal{T}, \mathcal{R})$). *If the $k \times k$ matrix L has (i, j)th element the loss from transmitter $i \in \mathcal{T}$ to receiver $j \in \mathcal{R}$ then*

$$\theta_{safe}(\mathcal{T}, \mathcal{R}) \leq \frac{1}{k-1} \tag{4.10}$$

Proof. (Partial proof.) For now, we show only that θ_{safe} as a function of the losses α_{ij}, $i \neq j$, has a local maximum wherever all the losses are equal, and θ_{safe} at such a point equals $1/(k-1)$. (It remains to be proved that these local maxima are global maxima.)

When all losses α_{ij}, $i \neq j$, are equal, the corresponding A-matrix equals $J_k - I_k$ for every matching, where J_k is the $k \times k$ matrix whose entries are all 1. $J_k - I_k$ has an eigenvector $(1, 1, \ldots, 1)^T$ with associated eigenvalue $k-1$. But any eigenvector with all positive entries must be associated with the Perron eigenvalue ([31], chapter I, pf. of thm. 4.1), so for every matching $\rho(A) = k - 1$ and by Theorem 10, $\theta_{safe} = 1/(k-1)$.

By changing any one of the losses α_{rs}, in the A-matrix associated with a matching in which node r interferes with node s, an off-diagonal entry increases above 1 while other entries remain 1, increasing the Perron eigenvalue for that matching. Since θ_{safe} depends on the Perron eigenvalue of the worst matching, $\theta_{safe}(\mathcal{T}, \mathcal{R})$ decreases below $1/(k-1)$. $\square$

The threshold $1/(k-1)$ was achieved when all losses were equal. Indeed, feasibility was shown in Theorem 1 to depend on keeping $\rho(A)$ small, and the Perron eigenvalue of a matrix is an increasing function of its entries. It follows that feasibility of various matchings will be limited by maximal elements of the corresponding A matrix.

Many bounds in this chapter derive from the following.

Theorem 12 (Frobenius (chapter II, theorem 1.1 of [31])). *Let X be a square matrix whose entries are nonnegative. Let $r(X) = \min_i \sum_j X(i,j)$ and $R(X) = \max_i \sum_j X(i,j)$ be the smallest and largest rowsums of X. The Perron eigenvalue of X lies in the range*

$$r(X) \leq \rho(X) \leq R(X) \tag{4.11}$$

The following theorem provides a lower bound on θ_{safe} to go with the upper bound of Theorem 11. Unlike in Theorem 11, these bounds depend on the α's.

Theorem 13 (lower bound on $\theta_{safe}(\mathcal{T}, \mathcal{R})$). *If the $k \times k$ matrix L has (i,j)th element the loss from transmitter $i \in \mathcal{T}$ to receiver $j \in \mathcal{R}$ then*

$$\theta_{safe}(\mathcal{T}, \mathcal{R}) \geq \frac{1}{\max_{r,s} \sum_{i \neq r} \frac{\alpha_{is}}{\alpha_{rs}}} \tag{4.12}$$

Proof. Let S_k denote the set of all permutations on k letters. $\{A(\pi) \mid \pi \in S_k\}$ is the set of $k!$ A-matrices associated with all matchings of receivers in $\mathcal{R}$ to transmitters in $\mathcal{T}$. Among the rows of these matrices, all k^2 row sums of the form $\sum_{i \neq r} \frac{\alpha_{is}}{\alpha_{rs}}$ appear. By Theorem 12 the Perron eigenvalue of A is upper bounded by its largest row sum R. So

$$\begin{aligned}\max_{\pi}\{\rho(A(\pi))\} &\leq \max_{\pi} \max_{i=1,\dots,k} R_i(A(\pi)) \\ &= \max_{r,s=1,\dots,k} \sum_{i\neq r} \frac{\alpha_{is}}{\alpha_{rs}} \qquad (4.13)\end{aligned}$$

Now the result follows from applying Theorem 10. □

Theorem 14. *For any set of losses, the spectral radius of $A(L,\pi)$ lies in the range*

$$\frac{1}{\max_i C(i,\pi i)} - 1 \leq \rho(A(L,\pi)) \leq \frac{1}{\min_i C(i,\pi i)} - 1. \qquad (4.14)$$

where C is the unique column stochastic matrix obtained from L by $C = LD$ where D is the diagonal matrix whose elements are the reciprocals of column sums of L.

Proof. The ith rowsum of $\mathscr{G}(C,\pi)$ is

$$\begin{aligned}\sum_{j=1}^{k} \mathscr{G}(C,\pi)(i,j) &= \sum_{j=1}^{k} \frac{C(j,\pi i)}{C(i,\pi i)} \qquad (4.15) \\ &= \frac{1}{C(i,\pi i)} \sum_{j=1}^{k} C(j,\pi i) \qquad (4.16) \\ &= \frac{1}{C(i,\pi i)} \qquad (4.17)\end{aligned}$$

By Theorem 12, $\rho(\mathscr{G}(C,\pi))$ lies between the smallest and largest of the rowsums of $\mathscr{G}$:

$$\min_i \frac{1}{C(i,\pi i)} \leq \rho(\mathscr{G}(C,\pi)) \leq \max_i \frac{1}{C(i,\pi i)} \qquad (4.18)$$

or

$$\frac{1}{\max_i C(i,\pi i)} \leq \rho(\mathscr{G}(C,\pi)) \leq \frac{1}{\min_i C(i,\pi i)} \qquad (4.19)$$

From Theorem 7, part 1, and Equation (3.30),

$$\rho(\mathscr{G}(C,\pi)) = \rho(\mathscr{G}(L,\pi)) = 1 + \rho(A(L,\pi)), \qquad (4.20)$$

which proves the theorem. □

Theorem 14 holds for all $\pi \in S_k$. We can extend it to

$$\max_{\pi \in S_k} \left\{ \frac{1}{\max_i C(i,\pi i)} - 1 \right\} \leq \max_{\pi \in S_k} \rho(A(L,\pi)) \leq \max_{\pi \in S_k} \left\{ \frac{1}{\min_i C(i,\pi i)} - 1 \right\}. \quad (4.21)$$

Taking reciprocals and employing Theorem 10 we get

$$\frac{1}{\max_{\pi \in S_k} \left\{ \frac{1}{\min_i C(i,\pi i)} - 1 \right\}} \leq \theta_{safe}(\mathcal{T},\mathcal{R}) \leq \frac{1}{\max_{\pi \in S_k} \left\{ \frac{1}{\max_i C(i,\pi i)} - 1 \right\}}. \quad (4.22)$$

or

$$\frac{1}{\left\{ \frac{1}{\min_{\pi \in S_k} \min_i C(i,\pi i)} - 1 \right\}} \leq \theta_{safe}(\mathcal{T},\mathcal{R}) \leq \frac{1}{\left\{ \frac{1}{\min_{\pi \in S_k} \max_i C(i,\pi i)} - 1 \right\}}. \quad (4.23)$$

Note that $\min_{\pi \in S_k} \min_i C(i,\pi i) = \alpha_{min}(C)$, the smallest element in the matrix C, so we have

Corollary 15.

$$\frac{1}{\frac{1}{\alpha_{min}(C)} - 1} \leq \theta_{safe}(\mathcal{T},\mathcal{R}) \leq \frac{1}{\left\{ \frac{1}{\min_{\pi \in S_k} \max_i C(i,\pi i)} - 1 \right\}}. \quad (4.24)$$

It might seem that the upper bound is expensive to compute. However, there is an algorithm which requires time polynomial in the size of the matching to compute it.

It turns out that the lower bound in Corollary 15 is identical to the lower bound of Theorem 13. No doubt this is because both were derived by applying the Theorem of Frobenius to the A matrix. We include both expressions because they differ in form.

Example Consider the matrices

$$L_1 = \begin{bmatrix} 0.039 & 0.013 & 0.020 \\ 0.014 & 0.009 & 0.012 \\ 0.144 & 0.012 & 0.011 \end{bmatrix} \tag{4.25}$$

$$L_2 = \begin{bmatrix} 0.14 & 0.015 & 0.015 & 0.022 & 0.015 & 0.008 \\ 0.016 & 0.055 & 0.1 & 0.07 & 0.054 & 0.2 \\ 0.3 & 0.02 & 0.032 & 0.047 & 0.02 & 0.012 \\ 0.011 & 0.065 & 0.04 & 0.05 & 0.065 & 0.8 \\ 0.053 & 0.122 & 0.072 & 0.204 & 0.144 & 0.06 \\ 0.04 & 0.01 & 0.048 & 0.05 & 0.01 & 0.009 \end{bmatrix} \tag{4.26}$$

The bounds from various theorems in this chapter on $\theta_{safe}(\mathcal{T}, \mathcal{R})$, as applied to L_1 and L_2, are listed in Table 4.1.

Table 4.1: Bounds on $\theta_{safe}(\mathcal{T}, \mathcal{R})$ for the example.

matrix	L_1	L_2
actual $\theta_{safe}(\mathcal{T}, \mathcal{R})$	.233	.0224
Prop. 11	$\theta_{safe} \leq .5$	$\theta_{safe} \leq .2$
Thm. 13	$.0765 \leq \theta_{safe}$	$.0074 \leq \theta_{safe}$
Cor. 15	$.0765 \leq \theta_{safe} \leq .36$	$.0074 \leq \theta_{safe} \leq .1272$

The theorems above bound θ_{safe} when specific sets of transmitters and receivers are given. We may be more interested in determining a safe value of θ for a network in which we are unsure which nodes will be receivers and which will be transmitters.

Theorem 16 (lower bound on $\theta_{safe}(k)$). *If all the losses in the network satisfy $\alpha_{min} \leq \alpha_{ij} \leq \alpha_{max}$ then*

$$\theta_{safe}(k) \geq \left(\frac{1}{k-1}\right)\frac{\alpha_{min}}{\alpha_{max}} \tag{4.27}$$

Proof. The largest possible row or column in the A-matrix corresponding to any matching between k transmitters and receivers has a single 0 and $k-1$ entries $\alpha_{max}/\alpha_{min}$. By Theorem 12 the Perron eigenvalue of A cannot exceed the sum of these entries. The result follows from Theorem 10. □

The following corollary to Theorem 16 follows from the fact that the largest matching in a network of N nodes has $k = \lfloor N/2 \rfloor$ pairs.

Corollary 17 (lower bound on $\theta_{safe}(*)$). *If all the losses in the network satisfy $\alpha_{min} \leq \alpha_{ij} \leq \alpha_{max}$ then*

$$\theta_{safe}(*) \geq \left(\frac{1}{\lfloor N/2 \rfloor - 1}\right)\frac{\alpha_{min}}{\alpha_{max}} \tag{4.28}$$

The bound is tight.

The lower bound of Corollary 17 is tight, in the sense that it is achieved by the following network of N nodes. Let the loss $\alpha_{i,\ i+\lfloor N/2 \rfloor} = \alpha_{min}$ for $i = 1, \ldots, \lfloor N/2 \rfloor$ while all other losses in the network are $\alpha_{ij} = \alpha_{max}$ where $j \neq i + \lfloor N/2 \rfloor$. Now the matching

$$1 \rightarrow 1 + \lfloor N/2 \rfloor,\ 2 \rightarrow 2 + \lfloor N/2 \rfloor, \ldots, \lfloor N/2 \rfloor \rightarrow 2\lfloor N/2 \rfloor \tag{4.29}$$

has an A-matrix

$$A = \begin{bmatrix} 0 & \frac{\alpha_{max}}{\alpha_{min}} & \cdots & \frac{\alpha_{max}}{\alpha_{min}} \\ \frac{\alpha_{max}}{\alpha_{min}} & 0 & \cdots & \frac{\alpha_{max}}{\alpha_{min}} \\ & \vdots & & \\ \frac{\alpha_{max}}{\alpha_{min}} & \frac{\alpha_{max}}{\alpha_{min}} & \cdots & 0 \end{bmatrix} \tag{4.30}$$

whose every rowsum equals the reciprocal of the right-hand side of inequality (4.28).

In applying these theorems to scheduling problems, we may find that one outlying node that is far from the others causes $\theta_{safe}(*)$ to be effectively zero. To avoid this, it may be useful to "exclude" nodes i whose $\max_j\{\alpha_{ij}, \alpha_{ji}\}$ lie below some threshold.

From the point of view of ad hoc network design, if the nodes are mobile, one may wish to move them in order to affect the losses α_{ij} so that different matchings become feasible.

If the nodes will be static, the communications system may be designed to ensure $\theta < \theta_{safe}$ under the expected propagation environment.

4.4 Appendix: Miscellaneous observations on the matrix $A(L, \pi)$

The following are interesting but were not directly useful in the flow of Chapters 3 and 4.

- The matrix A which corresponds to a matching is likely to be invertible because the matrix is composed of numbers representing a real-world propagation environment.

- For $k = 2$ the $k \times k$ matrix A is irreducible with index of imprimitivity $h = 2$. For $k \geq 3$ it can be seen by direct computation that A^2 has all entries positive. Therefore it is primitive ($h = 1$).

- $(1/\theta)I - A$ is an M-matrix. Therefore all its real eigenvalues are non-negative and its complex eigenvalues must have nonnegative real part. Every principal minor of it is nonnegative. Its inverse is nonnegative.

- In fact whenever the system is feasible Q^{-1} has the convergent series representation

$$\left(\frac{1}{\theta}I - A\right)^{-1} = \theta \sum_{0}^{\infty} (\theta A)^k \tag{4.31}$$

- The relation of $A(L, \pi)$ to the loss matrix L can be expressed as

$$A(\pi) = D(\pi, L)^{-1} P(\pi) L^T - I \tag{4.32}$$

where π is some permutation on k receivers, so that the matching

$$T_1 \to R_{\pi(1)}, T_2 \to R_{\pi(2)}, \ldots, T_k \to R_{\pi(k)} \tag{4.33}$$

will be feasible if and only if the the spectral radius of $A(\pi)$ is less than $1/\theta$; $P(\pi)$ is the permutation matrix which carries $(1, 2, \ldots, k)^T$ to $(\pi(1), \pi(2), \ldots, \pi(k))^T$ and $D(\pi, L)$ is the diagonal matrix whose entries are $D(i, i) = L(i, \pi(i))$.

Example If $L = \begin{bmatrix} \alpha_{11} & \alpha_{12} & \alpha_{13} & \alpha_{14} \\ \alpha_{21} & \alpha_{22} & \alpha_{23} & \alpha_{24} \\ \alpha_{31} & \alpha_{32} & \alpha_{33} & \alpha_{34} \\ \alpha_{41} & \alpha_{42} & \alpha_{43} & \alpha_{44} \end{bmatrix}$ and $\pi = (134)(2)$ then

$$A(\pi) = \begin{bmatrix} \alpha_{13}^{-1} & 0 & 0 & 0 \\ 0 & \alpha_{22}^{-1} & 0 & 0 \\ 0 & 0 & \alpha_{34}^{-1} & 0 \\ 0 & 0 & 0 & \alpha_{41}^{-1} \end{bmatrix} \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix} L^T - I_4 \tag{4.34}$$

$$= \begin{bmatrix} 0 & \frac{\alpha_{23}}{\alpha_{13}} & \frac{\alpha_{33}}{\alpha_{13}} & \frac{\alpha_{43}}{\alpha_{13}} \\ \frac{\alpha_{12}}{\alpha_{22}} & 0 & \frac{\alpha_{32}}{\alpha_{22}} & \frac{\alpha_{42}}{\alpha_{22}} \\ \frac{\alpha_{14}}{\alpha_{34}} & \frac{\alpha_{24}}{\alpha_{34}} & 0 & \frac{\alpha_{44}}{\alpha_{34}} \\ \frac{\alpha_{11}}{\alpha_{41}} & \frac{\alpha_{21}}{\alpha_{41}} & \frac{\alpha_{31}}{\alpha_{41}} & 0 \end{bmatrix} \tag{4.35}$$

The matrix equation relating A and L is interesting but not as useful as the elementwise descriptions of A and $\mathscr{G}$ in Equations (3.4) and (3.31). Perhaps this is because the matrix description hides some nonlinearities, such as computing the elements of D.

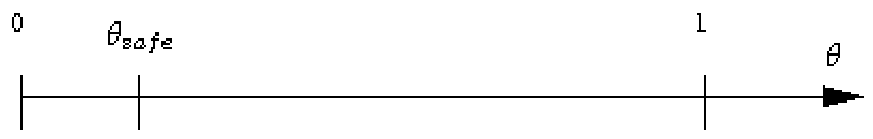


Figure 4.1: When $\theta > 1$, Theorem 9 says there can be no more than one feasible matching. When $\theta < \theta_{safe}$ all matchings are feasible.

Chapter 5

An asynchronous neighbor discovery algorithm for wireless sensor networks

5.1 Introduction

Neighbor discovery is the determination of all stations with which a station may communicate directly. It is an important and non-trivial task in wireless networks, particularly for sensors. In the case where sensors are immobile, it may make sense to pay a one time price to learn of the existence of all one's neighbors in order to optimize medium access and to enable routing.

There are several ways to discover neighbors. In one class of methods, there is a central controller. All stations report to the controller, which determines the positions of the stations, computes their neighbors, and informs each station. Central control of neighbor discovery is expected to cost a lot of energy, particularly when the number of nodes is large. Distributed algorithms have no central controller.

In the distributed algorithm of Baker and Ephremides [41], all nodes participate in a two-round round-robin schedule. In each round each station is assigned a single slot to announce its identity and the identities of neighbors discovered so far. Stations listen in the other slots, and can determine all their neighbors, and all their neighbors' neighbors, within two rounds, under the assumption that nodes receive messages from neighboring nodes without errors.

Although the algorithm of [41] is distributed, it requires global synchronization of the network. In particular, it assumes:

(G1) schedules may be formed among sets of ≥ 3 nodes, and

(G2) the nodes know ahead of time the time when the algorithm is supposed to begin.

The algorithm described in this chapter is also distributed, but has the additional advantage that it does not require global synchronization. When the algorithm concludes, each node has a (possibly incomplete) list of its neighbors.

The ability to work without global synchronization is useful for two reasons. First, neighbor discovery may be the very first algorithm run in a network. Subsequent network behavior may presume that some prior mechanism enforces synchronization, but for the first algorithm there is no basis for that presumption. Second, the burden of maintaining global synchronization through algorithmic means, or the expense of maintaining it through hardware such as GPS, would increase with the size of the network. A large wireless network implies a large expense to maintain global synchronization.

The usefulness of an asynchronous algorithm for wireless networks goes beyond the ability to compile a list of neighbors. The algorithm is a protocol for passing arbitrary messages between neighboring nodes. Some messages that could be passed are the exchange of messages in support of a transmission scheduling algorithm; the exchange of information about one's neighbors, eventually allowing several-hop-away neighbor information to be obtained, which may be used to support routing algorithms; the exchange of physical layer parameters such as received signal strength; and so on. It is even possible for this algorithm to carry messages which will allow for the subsequent synchronization of the network, thereby allowing algorithms which require synchronization to follow it.

5.2 Problem statement

Definition 10. *Node x is a neighbor of node y if x can exceed y's signal-to-noise-ratio requirement.*

We assume that:

1. each node can broadcast its transmission, or receive up to one broadcast at any time, but cannot simultaneously transmit and receive. Broadcasts from neighbors are received free of errors, provided only one neighbor is transmitting.

2. each node has and knows its own unique identifier.

3. all nodes have an estimate $\widehat{N}$ of the number of neighbors each is expected to find.

The requirement that each station have a unique identifier can be achieved in various ways. Each station might have a Network Interface Card ID, or a CPU ID, which it can access and whose uniqueness is assured by device manufacturers. Or, if the nodes are truly indistinguishable from a hardware point of view, a sufficiently large pool of numbers may be chosen from, such that it is highly unlikely two nodes near each other will collide in their choices. More elaborate schemes along this line are considered in [42].

The requirement for an estimate of the number of neighbors will often be satisfied in sensor networks. A known number of sensors may be deployed within a known area, and an estimate follows from experience of the radio range of the devices within the expected propagation environment.

Any neighbor discovery scheme requires there be a message m identifying the sender. We assume that m can be successfully transmitted in time T_m to a receiver. The receiver successfully receives m if no other station within distance d of the receiver is transmitting simultaneously. That is, transmissions colliding at a receiver are destroyed.

To counter the lack of global synchronization we include a preamble in the message m. It is therefore important that the receiver of m receive m from the beginning; no cut-ins are allowed.

Definition 11. *Node x discovers y if x and y are neighbors and x receives m from y at least once.*

Note that x may discover y without y discovering x.

Problem: Given a set of K immobile wireless nodes whose locations are unknown a priori, we seek a distributed algorithm, that does not require global synchronization of the network, and that maximizes the performance metric described below.

Our performance metric is the following. Let $\mathcal{A}$ be a neighbor discovery algorithm. Given a finite set of nodes, each node has a finite set of neighbors. There is a finite set of neighbor relations for the network, including both (x, y) and (y, x) if x and y are neighbors. Let

$$\mathcal{F}(t) = E\left\{\frac{\text{number of neighbor relations discovered by } \mathcal{A} \text{ in a period of length } t}{\text{actual number of neighbor relations}}\right\}, \tag{5.1}$$

where the right hand side is statistical expectation over possible locations of the nodes and offsets between nodes' clocks.

For the distributed neighbor discovery algorithm of [41], $\mathcal{F}(t) = 1$ if t is long enough to complete the first round of the schedule.

5.3 Algorithm description

We will describe a neighbor discovery algorithm $\mathcal{A}$.

We assume each station has its own timeslotting, with equal slot lengths, but at a random offsets to others. In each slot each node chooses among the states {T, R$\}$ corresponding to transmit and receive, with probabilities p_T and p_R satisfying $p_T + p_R = 1$. The nodes act independently of each other and choose their states independently in each slot.

We determine the slot lengths as follows. During a slot when a node has decided to transmit, it transmits W copies of message m, where W is a positive integer fixed throughout the network. In the case of slotted operation $W > 1$ is unnecessary, but for unslotted operation the optimal value of W remains to be determined.

Depending on the radio technology in use, there will be a maximum transmission rate and some time required to transit between states, which we assume is negligible. We assume that a period T_m is required to transmit m once, and that WT_m is required to transmit W copies of the message. Thus a single slot has duration $T = WT_m$.

During a receive slot, a node turns on its receiver and decodes its input. The node processes the input to determine whether an error-free message was received. If one was, then the identity of the transmitter is determined from the message contents, and if the transmitter was heretofore unknown to the receiver, he is added to a local list of neighbors. Any additional information in the message is stored or updated at the receiver.

To overcome the lack of an agreed start time for $\mathcal{A}$, we propose an additional algorithm $\mathcal{B}$, which is compatible with $\mathcal{A}$. $\mathcal{B}$ is described in Appendix 5.9. The effect of $\mathcal{B}$ is that nodes will begin to run $\mathcal{A}$ not all at once but at various times, such that neighbors' runs of $\mathcal{A}$ overlap.

5.4 Slotted analysis of $\mathcal{A}$

We begin by considering $\mathcal{A}$ in slotted time in this Section, then without slotting in Section 5.5. The purpose of this is to gain intuition, and produce some simple results that will be useful in the more complicated analysis of the next section.

Because synchronization is assumed, there is no benefit to repeating messages, so we let $W = 1$.

We assume that a period t has been fixed for the discovery of neighbors. This period includes $\mathcal{S} = \lfloor t/T_m \rfloor$ slots.

Let h be the number of successful receptions of m made by one node in one slot. Assume the parameters p_T, W and N are fixed. Then h is a random variable because the $N-1$ neighbors of the node in question are in unknown states during the slot. We are interested in computing $E(h)$.

Let a node X have $N-1$ neighbors. Since the N nodes act independently, and each timeslot is independent of the others, the slotted algorithm can be thought of as an $N \times \mathcal{S}$ table, in which each cell contains an independent identically distributed Bernoulli random variable. See Figure 5.1. Looking down any column of the table (one timeslot), we have receivers and transmitters. In any column, let R and T be the numbers of receivers and transmitters, respectively, among X's neighbors. Let h be the number of nodes heard by X in any slot. Clearly h cannot exceed 1, for if more than one neighbor of X transmits in a slot, X could only hear a collision. So h is a zero-one variable indicating whether exactly one of the neighbors of X is transmitting. The quantity $E(h)$ can be interpreted as the probability that X hears a neighbor Y in one timeslot. The expected number of hearings X makes is

$$\begin{aligned} E(h) &= Pr(X \text{ in state } \mathtt{R}\ )Pr(T=1) && (5.2)\\ &= p_R \cdot \binom{N-1}{1} p_T(1-p_T)^{N-2} && (5.3)\\ &= (N-1)p_T(1-p_T)^{N-1} && (5.4)\end{aligned}$$

Since all nodes act identically, we expect symmetry: of all the times node X hears another node who transmits alone, these are uniformly distributed among the his $N-1$ neighbors. Therefore the expected number of times that X hears Y, where Y is a particular neighbor of X, is $E(h)/(N-1)$.

The performance of the algorithm is the fraction of neighbors discovered in the whole network. The neighbor discovery performances of the various nodes are not independent. However if $\mathcal{S}$ is large the error in assuming independence will not be large.

The performance of a single node X can be determined by considering X's experience horizontally in Figure 5.1. Each timeslot is a trial in which a neighbor may be heard. When $E(h)/(N-1)$ is small and $\mathcal{S}$ large, we can think of the $\mathcal{S}$ slots as being a large number of trials with a small probability of success, and approximate the number of times X hears any neighbor by a Poisson variable with mean $E(h)/(N-1)$. In this case the probability that any node discovers any other is the probability that the Poisson variable is nonzero, or

$$\mathcal{F} = Pr(X \text{ discovers } Y) = 1 - e^{-\frac{\mathcal{S}E(h)}{N-1}} \qquad (5.5)$$

This equation is a key to the analysis. The designer controls $\mathcal{S}$ through W, and affects $E(h)$ through p_T. The right hand side of (5.5) is an increasing function

timeslot	1	2	3	4	5	6	7	8	$\cdots$	$\mathcal{S}$
node X	R	T	R	R	R	R	R	R	$\cdots$	R
neighbor 1	R	R	T	R	R	T	R	R	$\cdots$	R
neighbor 2	R	R	R	R	R	R	R	T	$\cdots$	R
neighbor 3	R	R	T	R	R	R	R	R	$\cdots$	R
$\vdots$	$\vdots$									
neighbor $N-1$	R	R	R	R	T	R	R	R	$\cdots$	T

Figure 5.1: An example. This chart depicts the slotted analysis of the neighbor discovery algorithm. Each of N rows describes the state sequence of one node. There are $\mathcal{S}$ columns, corresponding to the duration of the algorithm. Every symbol in the table represents the outcome of an independent identical experiment, because all nodes use the same p_T and $p_R = 1 - p_T$. Each column describes the state of the system in one timeslot. The contents of any column follow a binomial distribution. Node X successfully receives m from one of his $N-1$ neighbors if X is in state `R` and exactly one of the neighbors is in state `T`. This occurs in columns 5, 6, 8 and $\mathcal{S}$ (as long as none of nodes $5, \ldots, N-1$ are in state `T` to cause a collision.) Note that in column $\mathcal{S}$, X hears neighbor $N-1$, but this already happened earlier, so there is no benefit to it.

of $E(h)$. Consequently we want to choose p_T to maximize $E(h)$. Fixing $N = \widehat{N}$ and treating Equation (5.4) as a function of the single variable p_T (with $p_R = 1 - p_T$) we find the optimal value of p_T to be

$$p_T^* = 1/\widehat{N}. \tag{5.6}$$

This can be interpreted as a probabilistic analog of $\widehat{N}$ nodes taking turns transmitting.

A sensor network is in general a multihop environment. We must consider that each node is at the center of its own "disk," which contains a random number of nodes, not necessarily equal to $\widehat{N}$. When all nodes use $p_T = 1/\widehat{N}$, performance may be sub-optimal, because some nodes will experience $N < \widehat{N}$ and transmit with greater probability than $1/N$ (causing too many collisions), and others will experience $N > \widehat{N}$ and transmit with smaller probability than $1/N$ (leaving too many slots silent), where N is the actual number of nodes in a disk and $\widehat{N}$ is the network-wide average. However, neighbors will still be discovered. We examine this robustness issue further in Section 5.5.3.

5.5 Asynchronous analysis of $\mathcal{A}$

The primary goal of the asynchronous analysis of $\mathcal{A}$ is to determine the optimal values of W and p_T. The larger W is, the longer the slots are. Large W gives a better chance for the message m to be successfully received during a slot, but for a given period t over which $\mathcal{A}$ is to be run, large W reduces the number $\mathcal{S}$ of slots.

In this section we present all the ideas needed to analyze the asynchronous case. We relegate all the actual calculations to Appendix 5.8.

5.5.1 Model

Whereas in the slotted analysis h was a random variable because the states of the $N-1$ neighbors were random, in the asynchronous case the offsets of the neighbors' slottings are an additional source of randomness.

We again consider a station X who is listening during one slot $(0, T)$ where $T = WT_m$. Suppose X has $N-1$ neighbors. These neighbors have timeslots also of length T, but are offset randomly to X's. See Figure 5.2. Let x be an $(N-1)$-vector of offsets, distributed uniformly in the interval $(0, T)$, with x_i representing the offset of neighbor i with respect to X. We must consider two states for neighbor i, the "left" state of the neighbor in $(x_i - T, x_i)$ and the "right" state in $(x_i, x_i + T)$.

We break up the number of successful receptions per node per slot, h, as

$$h = h_{reg} + h_{rtc} + h_{mult} \tag{5.7}$$

where h_{reg} are regular hearings, h_{rtc} are round-the-corner hearings and h_{mult} are multiple hearings. To define these, let X be the node which will successfully receive a message m.

Definition 12. *A "regular hearing" occurs when X, in state* `R`*, receives m successfully during a single slot.*

Definition 13. *A "round-the-corner hearing" occurs when X, in state* `R` *for two*

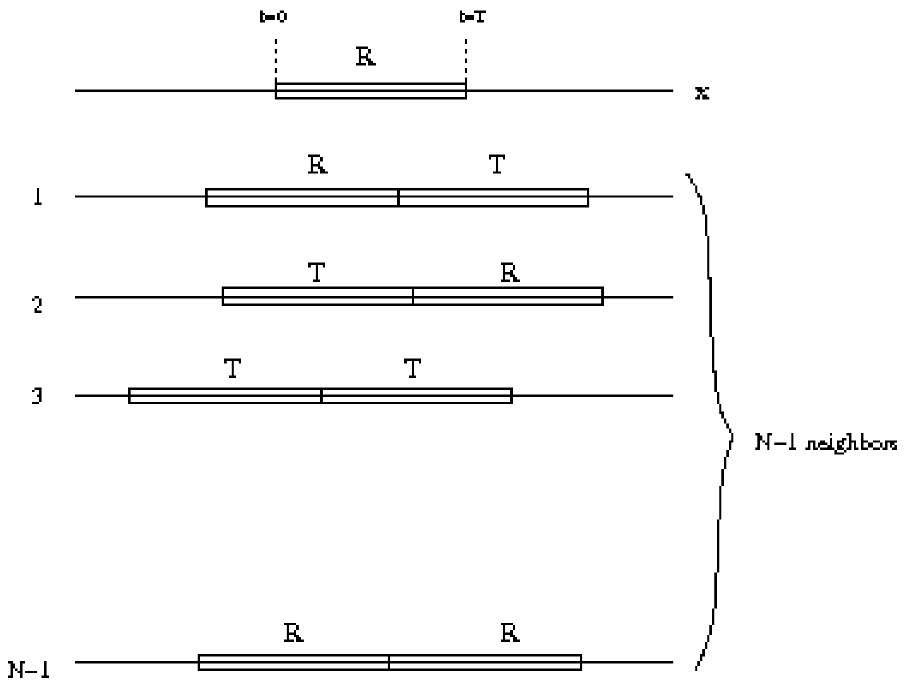


Figure 5.2: The reference timeslot (top line) of a wireless node X who is in state R, and the timelines of his $N-1$ neighbors. Each neighbor has two timeslots overlapping the slot of interest. The neighbors fall into four categories: double transmitters like neighbor 3 who transmit in both of their slots; left transmitters such as node 2; right transmitters such as node 1; and nodes who transmit in neither of the two overlapping slots, such as node $N-1$.

consecutive slots, begins receiving m in the first slot and finishes receiving it in the second slot.

Definition 14. *A "multiple hearing" occurs when X receives a second message in a single slot, whether from the same transmitter or a different one.*

Because all slots have length T, it is not possible for three or more messages to be received in a single slot. An example where two hearings can occur is the following. Let $W = 4$ and let X be in state R in the interval $(0, T)$. Node Y transmits in $(-2T/3, T/3)$ and X receives Y's final minislot. Z transmits in $(T/2, 3T/2)$ and X receives Z's first minislot. All other neighbors of X are in state `RR` in $(0, T)$.

Unless $W = 1$, most hearings are regular. A smaller number are round-the-corner. In this analysis we neglect multiple hearings. Multiple hearings have occurred only rarely in our experiments, and they are difficult to count. Our analysis therefore mildly under-estimates the performance of $\mathcal{A}$.

The detailed analysis of regular and round-the-corner hearings involves the careful analysis of sub-cases. These sub-cases will now be introduced. Evaluation of the expected numbers of hearings in these cases can be found in Appendix 5.8.

We start by observing that all of X's neighbors are in some pair of states, `TT`, `TR`, `RT` or `RR`, with respect to the slot in which X listens. We call these neighbors double transmitters, left transmitters, right transmitters and non-transmitters, respectively, and denote their numbers by D, L, R and $(N - 1) - (D + R + L)$.

X will hear nothing if $D \geq 2$. If $D = 0$, X could successfully receive m from a left transmitter or a right transmitter (but not both; that would be a multiple

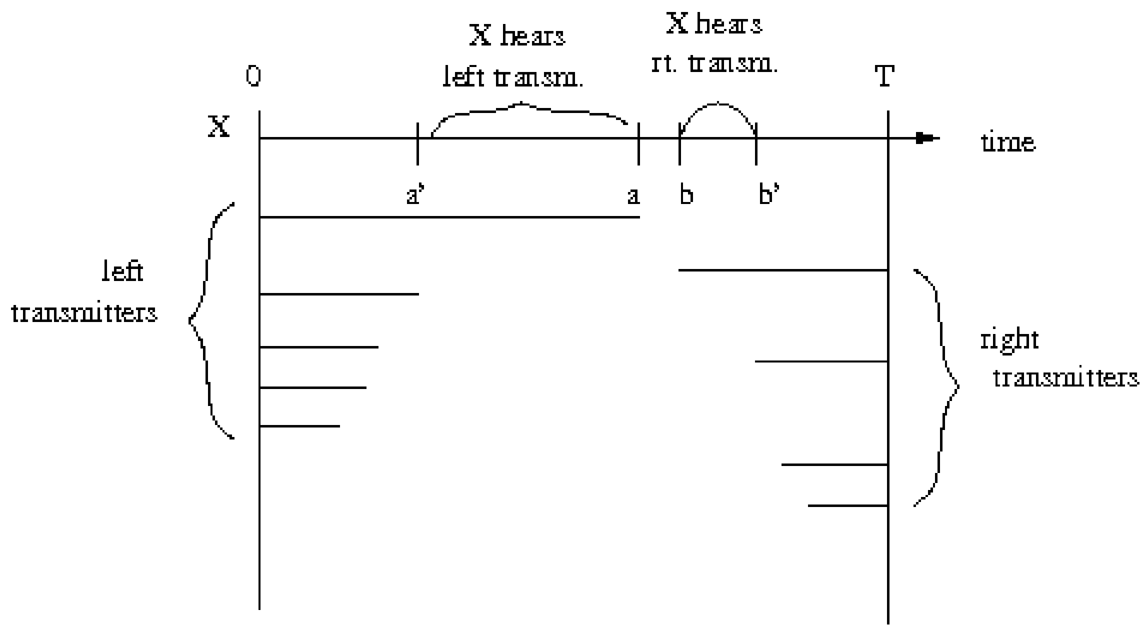


Figure 5.3: Regular hearing with $D = 0$, $a < b$. X can hear the left transmitter if $a - a' > T/W$ and can hear the right transmitter if $b' - b > T/W$.

hearing). There are two sub-cases in $D = 0$. One of the left transmitters has the largest offset in $(0, T)$, which we denote a. Another of the left transmitters has the second-greatest offset, a'. If $a - a'$ is greater than the length of a minislot, and no right transmitter interferes, then the left transmitter with offset a succeeds in getting his message to X. This situation, and its analogy for right transmitters is depicted in Figure 5.3.

Even if the left transmitter and the right transmitter overlap, as in Figure 5.4, one of them can succeed. If the period from a' to b contained an entire message from a left transmitter, X would receive it. If the width of the gap $b - a'$ is one minislot or less, m is not received. If $b - a'$ is the length of two minislots or more, success is guaranteed. If $b - a'$ is the length of one and a half minislots or more, we

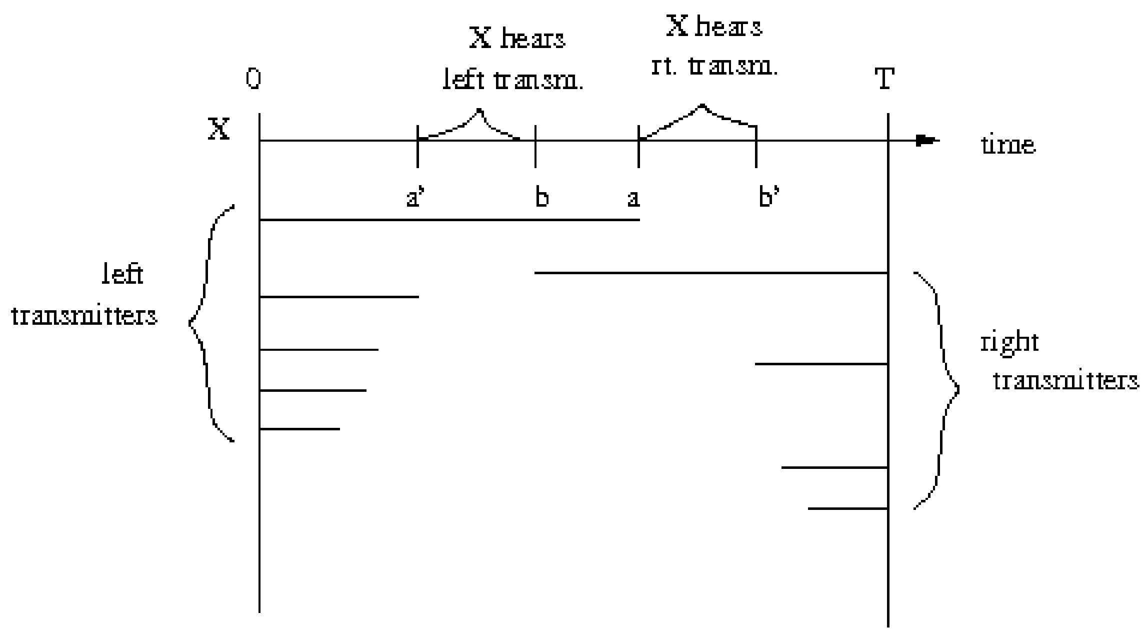


Figure 5.4: Regular hearing with $D = 0$, $a > b$. X can hear the left transmitter, on average, if $b - a' > (3/2)T/W$ and can hear the right transmitter, on average, if $b' - a > (3/2)T/W$.

assume for simplicity that the transmission succeeds.

When $D = 1$, as in Figure 5.5, X could only hear the double-transmitter. X does so if the left and right transmitters leave a sufficiently large gap for him, i.e. if $b - a$ is large enough. Again, a one minislot gap is not enough because of X's need to hear the preamble; a two minislot gap would guarantee success. On average, because the double transmitter's offset is distributed uniformly on $(0, T)$ we count a gap of one and a half minislots in length or greater as a success, and shorter as a failure.

The round-the-corner situation is depicted in Figure 5.6. Let Y be a node which has a transmit slot overlapping two receive slots of X. Some transmit minislot

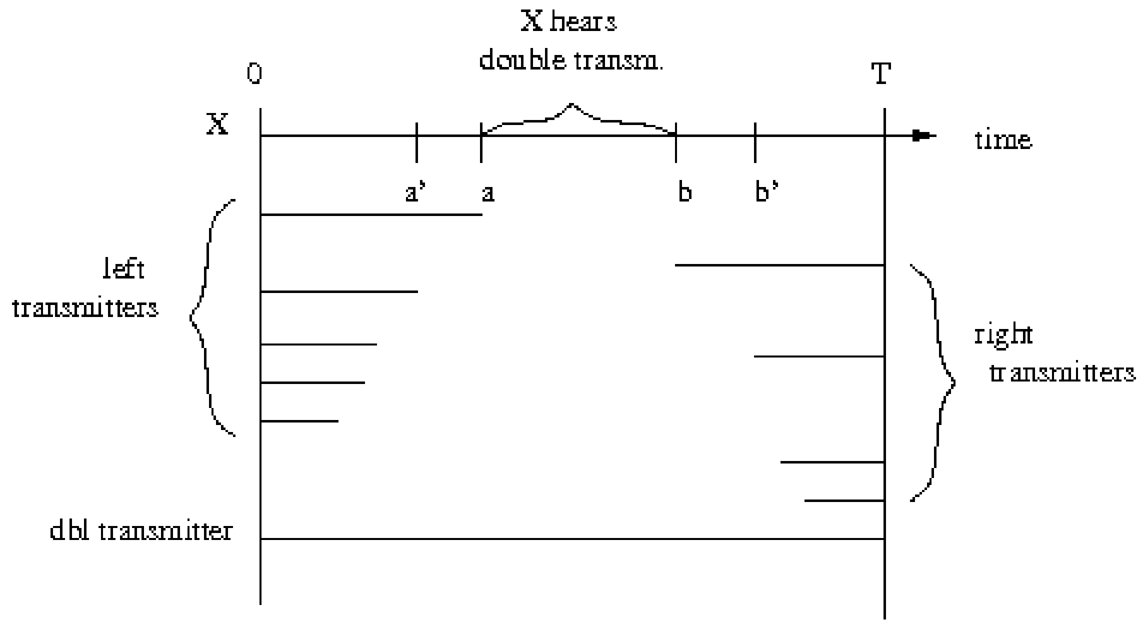


Figure 5.5: Regular hearing with $D = 1$, $a < b$. X can hear the double transmitter, on average, if $b - a > (3/2)T/W$.

of Y straddles X's slots. If during that minislot no other neighbor of X transmits, X makes a round-the-corner hearing. Each other neighbor of X has two minislots which overlap Y's. The probability of success is then the probability that all these $2(N-2)$ minislots are silent.

5.5.2 Performance of $\mathcal{A}$

Recall (5.5):

$$\mathcal{F} = 1 - e^{-\frac{\mathcal{S}E(h)}{N-1}}$$

This remains true for the asynchronous case, provided that $E(h)$ is computed in a way appropriate to an unslotted system, $\mathcal{S}$ is large and $\mathcal{S}E(h)/(N-1)$ is small.

When $t \gg WT_m$, we have $\mathcal{S} = \lfloor t/(WT_m) \rfloor \approx t/(WT_m)$. The performance

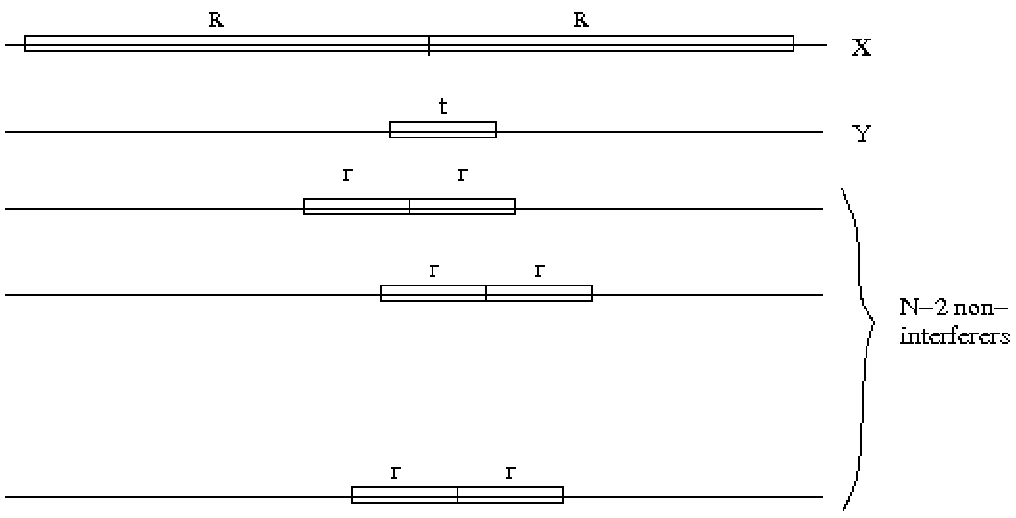


Figure 5.6: Round-the-corner hearings. X receives for two consecutive slots. Y transmits in the minislot overlapping the two. The other neighbors of X have two quiet minislots overlapping Y's transmit minislot.

metric can be expressed as

$$\mathcal{F} \approx 1 - e^{-\frac{t}{T_m}\frac{E(h)}{W}\frac{1}{N-1}} \tag{5.8}$$

Performance increases monotonically with the exponent. The first factor t/T_m is given. The third factor can be estimated. Therefore performance really depends on the ratio $E(h)/W$. We will use $E(h)/W$ as a figure of merit for W and p_T.

In Appendix 5.8 estimates of $E(h)/W$ for various values of N, W and p_T are derived. They are plotted in Figure 5.7. In this figure the light colored blocks have better $E(h)/W$. Unless $N = 2$ (which is not likely to be planned, since a network in which the average number of neighbors is 1 is very sparse and likely to be disconnected) it appears that $W = 2$ is the best choice for the asynchronous operation of $\mathcal{A}$.

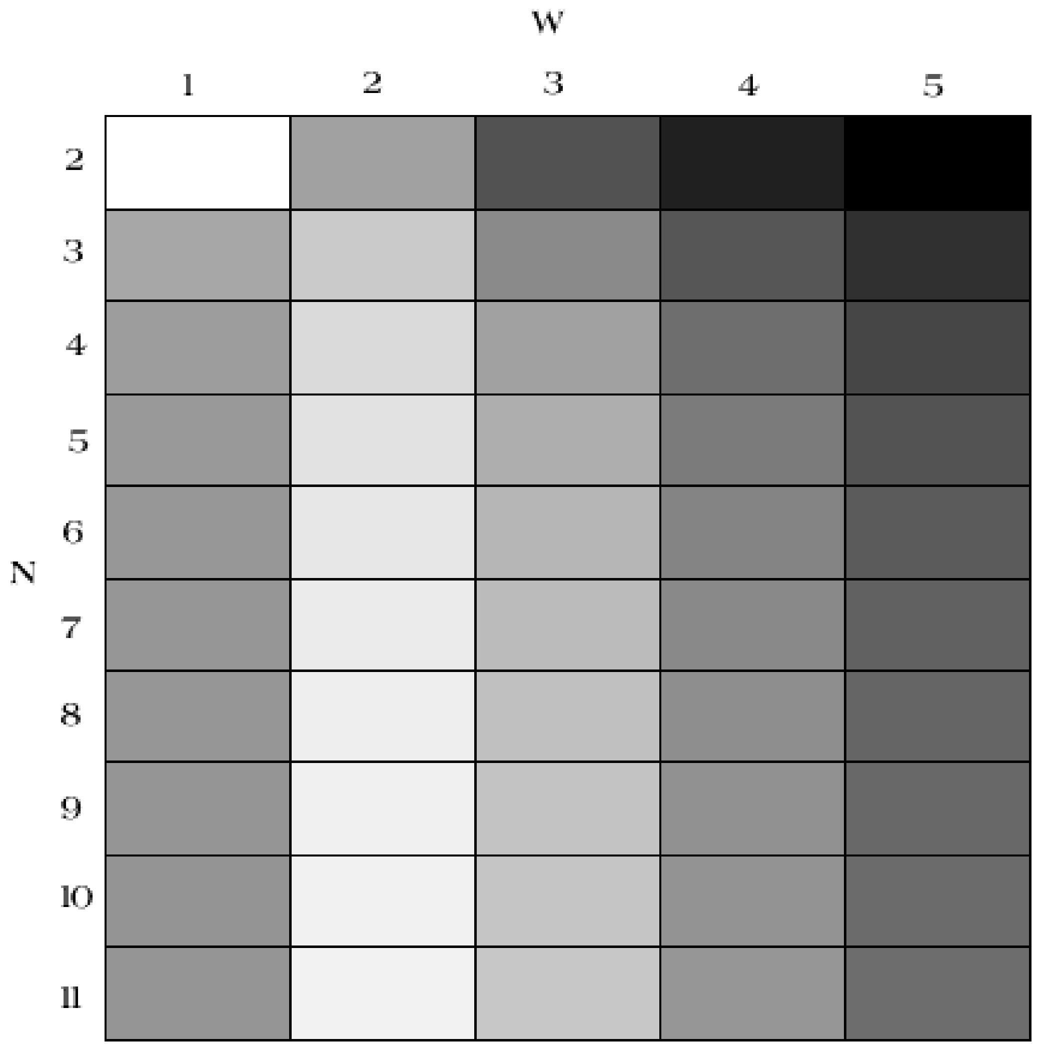


Figure 5.7: A density plot of $E(h)/W$. Lighter blocks have greater $E(h)/W$. The values for $E(h)$ are taken from Table 5.2.

5.5.3 Robustness of $\mathcal{A}$

In the previous sections we worked out the optimal W and p_t given N. In place of N we suggested that the network-wide average number $\widehat{N}$ be used. However for most nodes in the network, the actual number of neighbors will not equal the average number of neighbors. In this section we examine how this affects performance, and consider whether we should use p_T different from the one expected to maximize performance in the average case.

Example Consider a network with $\widehat{N} = 6$, $W = 2$, and $\mathcal{S} = 100$ slots. In Appendix 5.8 we determine that the optimal p_T in this case equals .12.

Figure 5.8 compares the performance of $p_t \in \{.10, .11, .12, .13, .14\}$ in the cases $N = 2, 3, \ldots, 11$.

When $N < 6$, the optimal transmission probability is greater than .12. When $N > 6$ the optimal transmission probability is smaller than .12.

It does not appear that choices of $p_T \neq .12$ offer significant advantages, unless more is known about the distribution of neighbors.

If one knew the complete distribution of numbers of neighbors (e.g. two-dimensional Poisson with a given mean), instead of just the mean, then it would be straightforward to choose a transmission probability which maximized performance over all the possible numbers of neighbors.

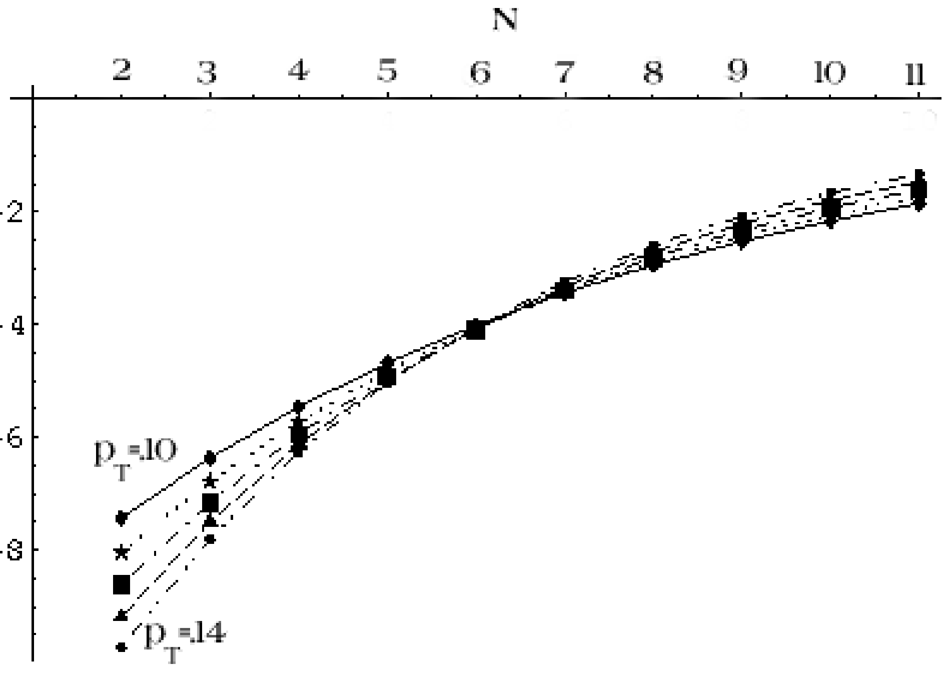


Figure 5.8: The expected fraction of undiscovered neighbors $\log_{10}(1-\mathcal{F})$ as a function of the actual number of neighbors, for five values of p_T.

5.6 Discussion

We have assumed that if two neighbors simultaneously transmit their messages, X hears garbage during the period of their overlap; X must hear exactly one of his neighbors transmitting a complete message. If multi-user detection were possible, then more transmissions would be successfully received and the performance of this algorithm would be improved.

The algorithm produces diminishing returns as it runs. At first, every reception of the message m from a neighbor is new; as the algorithm runs and more messages are received, more of them duplicate messages already received; finally, a node may continue to run $\mathcal{A}$ well after it has (unknowingly) discovered all its neighbors. If $\mathcal{A}$ runs long enough, a majority of the energy used by $\mathcal{A}$ is wasted, since only the first time one transmits successfully to a neighbor is one getting useful information across. Later successful transmissions to that neighbor are redundant.

In this chapter it was assumed that the running time t is fixed. If instead we were to choose a stopping point, we could use Equation (5.5) as a basis for determining performance as a function of time.

The nodes might adapt local parameter settings based on observations. During $\mathcal{A}$, they listen during every slot they do not transmit. If a nodes hears few transmissions, from the fact that p_T is common across the network, it could conclude that it has few neighbors. If this were the case, a higher p_T might be called for. Performance might be improved by simple adaptive behavior such as this.

5.7 Conclusion

Neighbor discovery is an important task in sensor networks. This chapter has presented the first asynchronous, distributed neighbor discovery algorithm.

The algorithm $\mathcal{A}$ (and an associated algorithm $\mathcal{B}$ described in Appendix 5.9) impose a state machine structure on the participating nodes and promise simple operation. Furthermore, the probabilistic nature of $\mathcal{A}$ makes it robust in the event that the actual number of neighbors is different (even quite different) from what was expected.

The algorithm's performance was analyzed in slotted and unslotted time. Optimal values of the settable parameters, p_T and W were derived, and expressions of performance, namely the number of successful receptions of the message m per node per timeslot, were provided as functions of N, p_T and W.

It remains to compare the performance of $\mathcal{A}$ with other neighbor discovery algorithms such as that of [41]. A major difference between the algorithms is that $\mathcal{A}$ is probabilistic, so not all neighbor relations are necessarily discovered. However, simulations indicate that $\mathcal{A}$ can discover nearly all neighbors with proper parameter settings.

5.8 Appendix: Asynchronous analysis of $E(h)$

In this appendix we represent $E(h)$ as a function of W, to enable us to choose an optimal value of W. As explained in Section 5.5, we consider a station X which is in state `R` and determine the probability that he successfully receives from one of his $N-1$ neighbors.

The length of a slot is T. It will be convenient to divide every slot into W minislots of length T/W. By analogy to `T` and `R` we will denote a node's state in a minislot by `t` and `r` for transmit and receive, respectively.

5.8.1 Regular hearings

In this section we derive the number of "regular hearings" X can expect to make per slot in which he is in state `R`, as a function of N, W and p_T.

We divide the neighbors into four categories. The neighbors whose state pairs are `TT` are double transmitters. The neighbors whose state pairs are `TR` are left transmitters. The neighbors whose state pairs are `RT` are right transmitters. The neighbors whose state pairs are `RR` are non-transmitters. We denote their numbers by D, L, R, and $N-1-(D+R+L)$ respectively. This categorization is sufficient state information for our purposes. Because all states are independent, the state probabilities are tetranomial with

$$\begin{aligned} Pr(D,R,L \mid N) &= \binom{N-1}{D,R,L}(p_T^2)^D\,(p_Rp_T)^R\,(p_Tp_R)^L\,\left(p_R^2\right)^{N-1-(D+R+L)} \\ &= \binom{N-1}{D,R,L}p_T^{2D+R+L}(1-p_T)^{2(N-1)-(2D+R+L)} \end{aligned} \tag{5.9}$$

We will combine the effect of the left transmitters as follows. Let a and a' be the greatest and second-greatest elements of the set

$$x_L = \{x_i : i \text{ is a left transmitter}\} \tag{5.10}$$

with $a' = 0$ if the set is a singleton and $a = a' = 0$ if empty. Similarly let b and b'

be the smallest and second-smallest elements of the set

$$x_R = \{x_i : i \text{ is a right transmitter}\} \tag{5.11}$$

with $b' = T$ if the set is a singleton and $b = b' = T$ if empty. Since x is a random variable, a, a', b and b' are order statistics. Assuming the x_i are uniformly distributed on $(0, T)$, and that there are $L \geq 1$ left transmitters and $R \geq 1$ right transmitters, the probability density functions of a and b, and the cumulative density functions of $a' \mid a$ and $b' \mid b$ are

$$f_a(a) = (L/T)(a/T)^{L-1} \quad 0 \leq a \leq T \tag{5.12}$$

$$f_b(b) = (R/T)(1 - b/T)^{R-1} \; 0 \leq b \leq T \tag{5.13}$$

$$F_{a'|a}(a') = \begin{cases} 0 & a' < 0 \\ (a'/a)^{L-1} & 0 \leq a' \leq a \\ 1 & a' > a \end{cases} \tag{5.14}$$

$$F_{b'|b}(b') = \begin{cases} 0 & b' < b \\ 1 - \left(\frac{T-b'}{T-b}\right)^{R-1} & b \leq b' \leq T \\ 1 & b' > T \end{cases} \tag{5.15}$$

As stated in Section 5.2, X must receive the entire message m from some transmitter while no other node in his neighborhood is transmitting. It is not enough for X to receive from some neighbor for a period of $T_m = T/W$ or longer; X must receive for T/W beginning with the preamble.

We'll now derive necessary and sufficient conditions for a regular hearing. Figures 5.3, 5.4 and 5.5 illustrate the three sub-cases of regular hearings.

X will be unable to hear any neighbor if there are two or more double transmitters. Thus we need only consider $D = 0$ and $D = 1$.

In the case $D = 0$, X may only hear the left transmitter whose offset is maximal (a), or the right transmitter whose offset is minimal (b), since the other transmitters are always interfered with by these two in $(0, T)$. If $a < b$ then if $a - a' > T/W$ a left transmitter is heard, since his final transmission minislot reaches X without a collision. If $b' - b > T/W$ a right transmitter is heard, since his first transmission minislot reaches X without a collision.

$$\begin{aligned} E(h_{reg} \mid a, b, a < b) &= 1 - Pr\left(a - a' < \frac{T}{W}\right) Pr\left(b' - b < \frac{T}{W}\right) \\ &= 1 - Pr\left(a' > a - \frac{T}{W}\right) Pr\left(b' < b + \frac{T}{W}\right) \\ &= 1 - \left(1 - F_{a'|a}\left(a - \frac{T}{W}\right)\right) F_{b'|b}\left(b + \frac{T}{W}\right) \qquad (5.16) \end{aligned}$$

in which the first equality follows from the independence of the offsets.

If $a > b$, it is still possible for X to hear the left transmitter if $b - a' > T/W$ or the right transmitter if $b' - a > T/W$. The probability of successful reception equals the probability that an entire minislot of the transmitter fits in the opening. This probability is zero if the opening has the width of one minislot (T/W) and one if the opening has the width of two minislots $(2T/W)$. We count a success when the width of the opening is $(3/2)T/W$, where the probability that the opening contains an entire minislot is 1/2.

$$\begin{aligned}
E(h_{reg} \mid a, b, a > b) &= 1 - Pr\left(b - a' < \frac{3}{2}\frac{T}{W}\right) Pr\left(b' - a < \frac{3}{2}\frac{T}{W}\right) \\
&= 1 - Pr\left(a' > b - \frac{3}{2}\frac{T}{W}\right) Pr\left(b' < a + \frac{3}{2}\frac{T}{W}\right) \\
&= 1 - \left(1 - F_{a'|a}\left(b - \frac{3}{2}\frac{T}{W}\right)\right) F_{b'|b}\left(a + \frac{3}{2}\frac{T}{W}\right) \quad (5.17)
\end{aligned}$$

Using these facts we may express

$$\begin{aligned}
E(h_{reg} \mid D = 0, R, L) &= \int_{a:0}^{T}\int_{b:0}^{T} E(h_{reg} \mid a, b) f_a(a) f_b(b) dbda \\
&= \iint_{a<b} E(h_{reg} \mid a, b, a < b) f_a(a) f_b(b) dbda \\
&\quad + \iint_{a>b} E(h_{reg} \mid a, b, a > b) f_a(a) f_b(b) dbda \\
&= 1 - \iint_{a>b} F_{b'|b}\left(a + \frac{3}{2}\frac{T}{W}\right) f_a(a) f_b(b) dadb \quad (5.18) \\
&\quad + \iint_{a>b} F_{a'|a}\left(b - \frac{3}{2}\frac{T}{W}\right) F_{b'|b}\left(a + \frac{3}{2}\frac{T}{W}\right) f_a(a) f_b(b) dadb \\
&\quad - \iint_{a<b} F_{b'|b}\left(b + \frac{T}{W}\right) f_a(a) f_b(b) dadb \\
&\quad + \iint_{a<b} F_{a'|a}\left(a - \frac{T}{W}\right) F_{b'|b}\left(b + \frac{T}{W}\right) f_a(a) f_b(b) dadb
\end{aligned}$$

This expression turns out not to depend on T.

We now consider the case $D = 1$. See Figure 5.5. X can only hope to hear the double transmitter. The double transmitter has a uniformly distributed offset. By an argument similar to the one made above, we count a success when the gap $b - a$ between the right and left transmitters exceeds $(3/2)T/W$. Given L and R, the expected number of hearings X makes is

$$\begin{aligned} E(h_{reg} \mid D=1, L, R) &= Pr\left(b-a > \frac{T}{W}\right) \\ &= \iint_{b-a>\frac{3}{2}\frac{T}{W}} f_a(a) f_b(b) db da \\ &= \begin{cases} 1 & L=0, R=0 \\ \frac{L!R!}{(L+R)!}\left(1-\frac{3}{2}\frac{1}{W}\right)^{R+L} & L+R \geq 1 \end{cases} \end{aligned} \tag{5.19}$$

a result which can be produced by repeated integration by parts.

We can average over the numbers of left, right and double transmitters to produce, for $W \geq 2$,

$$E(h_{reg}) = \sum_{D=0,1} \sum_{\substack{L,R \\ 0 \leq L+R \leq N-1-D}} Pr(D, L, R) E(h_{reg} \mid D, L, R) \tag{5.20}$$

whose summand can be expanded by the substitution of (5.18), (5.19) and (5.9). Examination of (5.18) and (5.19) reveals that $E(h_{reg}) = 0$ when $W = 1$.

5.8.2 Round-the-corner hearings

In this section we determine the expected number of round-the-corner hearings we expect to make. Specifically, let X again be a listening station but now listening for two consecutive slots. Let some neighbor Y transmit in a slot which overlaps the two listening slots.

In order to avoid double-counting regular hearings, we only want to consider the particular minislot in which Y transmits and which is heard at the end of X's first slot and completes in X's second slot.

The other $N-2$ neighbors of X must all be in state `R` for both of their minislots which overlap Y's minislot. Refer to Figure 5.6.

Let B be the event that a boundary between minislots is also a boundary between slots. Then the probability that a node is in state `rr` during a pair of minislots is

$$\begin{aligned} Pr(\texttt{rr}) &= Pr(\texttt{rr}|B)Pr(B) + Pr(\texttt{rr}|B^C)(1 - Pr(B)) && (5.21)\\ &= Pr(\texttt{RR})\left(\frac{1}{W}\right) + Pr(\texttt{R})\left(1 - \frac{1}{W}\right) && (5.22)\\ &= p_R^2\left(\frac{1}{W}\right) + p_R\left(1 - \frac{1}{W}\right) && (5.23)\end{aligned}$$

Therefore, given that X is in state `RR`, the probability that X successfully receives a round-the-corner hearing is

$$E(h_{rtc}) = \binom{N-1}{1} p_T \left(p_R^2\frac{1}{W} + p_R\left(1 - \frac{1}{W}\right)\right)^{N-2} \qquad (5.24)$$

5.8.3 Optimal settings

We may combine the results of the previous two subsections to see for various values of N, W, and p_T how many hearings we expect. Because regular hearings can only be made by a node which is listening, and round-the-corner hearings can only be made by a node listening in two consecutive slots, over a period of $\mathcal{S}$ slots we would expect to see

$$\mathcal{S}p_R E(h_{reg}) + (\mathcal{S} - 1)\, p_R^2 E(h_{rtc}) \qquad (5.25)$$

successful receptions per node. Per slot, each node would receive

$$\begin{aligned} E(h) &= p_R E(h_{reg}) + \frac{\mathcal{S}-1}{\mathcal{S}} p_R^2 E(h_{rtc}) && (5.26)\\ &\approx p_R E(h_{reg}) + p_R^2 E(h_{rtc}) && (5.27)\end{aligned}$$

for large $\mathcal{S}$.

Table 5.1: Optimal transmit probability as function of N and W, to nearest .01.

N	$W = 1$	$W = 2$	$W = 3$	$W = 4$	$W = 5$
2	.50	.42	.41	.40	.40
3	.22	.26	.28	.28	.29
4	.15	.19	.21	.21	.22
5	.12	.15	.16	.17	.18
6	.09	.12	.14	.14	.15
7	.08	.10	.12	.12	.13
8	.07	.09	.10	.11	.11
9	.06	.08	.09	.10	.10
10	.05	.07	.08	.09	.09
11	.05	.06	.07	.08	.08

For selected values of N and W we have computed the $\widehat{p}_T$ which maximizes $E(h)$. These values, in Table 5.1, were computed by exhausting over p_T in the set $\{.01, .02, \ldots, .99\}$ using Equation (5.27). Recall that in the slotted case, $\widehat{p_T} = 1/N$. With asynchrony this is evidently no longer true.

Table 5.2 shows the expected number of hearings that could be achieved on average, per node which has $N - 1$ neighbors. The figures are produced from Equation (5.27) using the transmit probabilities of Table 5.1.

Figure 5.7 is derived from these data, and shows that $W = 2$ is the best choice except for the sparsest networks.

Table 5.2: Values of $E(h)$ which result from using the transmit probabilities of Table 5.1.

N	$W = 1$	$W = 2$	$W = 3$	$W = 4$	$W = 5$
2	0.250	0.385	0.432	0.456	0.470
3	0.196	0.434	0.535	0.587	0.617
4	0.190	0.454	0.578	0.645	0.685
5	0.187	0.464	0.601	0.677	0.724
6	0.186	0.471	0.615	0.698	0.749
7	0.186	0.475	0.625	0.712	0.767
8	0.186	0.478	0.633	0.723	0.779
9	0.185	0.481	0.639	0.730	0.789
10	0.185	0.483	0.643	0.737	0.797
11	0.185	0.483	0.646	0.743	0.803

5.9 Appendix: A distributed algorithm to commence neighbor discovery

In this appendix we describe a distributed algorithm $\mathcal{B}$ which relieves the network of having to arrange for all the nodes to know and agree upon a common start time for $\mathcal{A}$, the neighbor discovery algorithm.

Whereas in the description of $\mathcal{A}$ all nodes could be in two states, `T` or `R`, for algorithm $\mathcal{B}$ we add a third state, `S` (for Sleep). Nodes enter these states independently as in $\mathcal{A}$ with probabilities p_T, p_R and p_S where $p_T + p_S + p_R = 1$.

In $\mathcal{B}$, most nodes are deployed in the following initial state, called "listen-only" mode

$$(p_T = 0, p_R = \epsilon, p_S = 1 - \epsilon) \tag{5.28}$$

where $\epsilon \ll 1$. The rest are deployed in "discovery" mode, defined by

$$(p_T, p_R = 1 - p_T, p_S = 0) \tag{5.29}$$

with $p_T > 0$ being whatever value has been selected for algorithm $\mathcal{A}$. To be in discovery mode means nothing more than to run algorithm $\mathcal{A}$. A node transits from listen-only mode to discovery mode only when it is in state `R` and receives a message on the shared channel (or has the incoming energy threshold exceeded, suggesting a collision) from another station. Once in discovery mode, a node remains in it for a fixed duration of $\mathcal{S}$ timeslots. After the period in discovery mode, the node has a possibly incomplete list of its neighbors. After running $\mathcal{A}$, the node could carry out the next task (e.g. scheduling, routing, data transfer) it has been programmed to do.

As an alternative to deploying some nodes in discovery mode, and others in listen-only mode, all nodes could be programmed with a countdown timer which forces a transition from listen-only to discovery mode if the transition has not already been made.

All nodes can be programmed the same way. The nodes differ only in their initial states and in their identities. The nodes do not need to agree on when $\mathcal{A}$ is "supposed to begin," because their transition from listen-only to discovery mode is event-triggered. Entry into $\mathcal{A}$ could be begun by any node, or by multiple nodes, with the same network behavior resulting. There will be a wave of nodes commencing $\mathcal{A}$ centered at the first node(s) who entered it.

5.9.1 Setting parameters for listen-only mode

It should be clear from the description of $\mathcal{B}$ that there is a nonzero probability that a node may remain in listen-only mode forever, never hearing any of his neighbors transmit during their discovery modes. As this could cause an entire subgraph behind the unlucky node to go undiscovered, it is important to set ϵ (probability of listening in listen-only mode) such that the probability of this happening be small. If the period preceding the beginning of discovery mode were expected to be short then we should consider making ϵ large; on the other hand if the deployment period is long, saving energy in it becomes worthwhile so we are reluctant to make it large.

Example Suppose we set as our goal that the probability that a station remains in listen-only mode for more than $\mathcal{S}/10$ slots while any one neighbor is in discovery mode be no more than δ. This probability is geometric with probability of the undesired event being $(1-\epsilon p_T)^{\mathcal{S}/10} \leq \delta$. Therefore we should set $\epsilon \geq (1-\delta^{10/\mathcal{S}})/p_T$ to guarantee the condition.

With ϵ set as above the probability that a node will completely miss a neighbor's entire discovery mode is δ^{10}. The expected delay between the time a node commences discovery mode and the time a neighbor detects this, also depends on ϵ. It is $1/(\epsilon p_T) = 1/(1-\delta^{10/\mathcal{S}})$. Both these situations improve when a node has more than one neighbor.

Of course, when a node has no neighbors, no neighbor discovery algorithm could succeed, and the node has no value to the sensor network.

BIBLIOGRAPHY

[1] H. Zimmerman. OSI reference model - the ISO model of architecture for open systems interconnection. *IEEE Trans. on Communications*, COM-28(4):425–432, Apr. 1980.

[2] S. Grandhi, R. Vijayan, D. Goodman, and J. Zander. Centralized power control in cellular radio systems. *IEEE Trans. Vehicular Technology*, 42(4):466, November 1993.

[3] S. A. Borbash and A. Ephremides. Energy, routing and decentralized detection in a wireless sensor network. In *Proc. NATO Workshop on cross-layer design principles*, 2004.

[4] S. A. Borbash and A. Ephremides. Wireless link scheduling with power control. In *Proc. WiOpt*, 2004.

[5] S. A. Borbash and A. Ephremides. The feasibility of matchings in a wireless network. In *Proc. Intl. Symp. Information Theory*, 2004.

[6] M. McGlynn and S. Borbash. Birthday protocols for low energy deployment and flexible neighbor discovery in ad hoc wireless networks. In *Proc. MobiHoc*, 2001.

[7] S. Tilak, N. B. Abu-Ghazaleh, and W. Heinzelman. A taxonomy of wireless micro-sensor network models. *Mobile Computing and Comm. Review*, 6(2):28–36, April 2002.

[8] A. Ephremides. Energy concerns in wireless networks. *IEEE Wireless Communications*, 9(4):48–59, August 2002.

[9] R. Viswanathan and P. K. Varshney. Distributed detection with multiple sensors: Part I- fundamentals. *Proc. IEEE*, 85(1), January 1997.

[10] J. Tsitsiklis. *Advances in Statistical Signal Processing*, volume 2, chapter Decentralized Detection, pages 297–344. JAI Press Inc, 1993.

[11] Pramod K. Varshney. *Distributed Detection and Data Fusion*. Springer, 1997.

[12] J. Tsitsiklis. Decentralized detection by a large number of sensors. *Mathematics of Control, Signals and Systems*, 1:167–182, 1988.

[13] P. Chen and A. Papamarcou. New asymptotic results in paprallel distributed detection. *IEEE Trans. Information Theory*, 39:1847–1863, Nov. 1993.

[14] W. W. Irving and J. N. Tsitsiklis. Some properties of optimum thresholds in decentralized detection. *IEEE Trans. Auto. Control*, 39:835–838, April 1994.

[15] R. Tenney and N. Sandell. Detection with distributed sensors. *IEEE Trans. on Aerospace and Electronic Systems*, 17(4), July 1981.

[16] L. Yu and A. Ephremides. Detection performance and energy efficiency trade-off in a sensor network. In *Forty-First Annual Allerton Conference On Communication, Control, And Computing*, October 2003.

[17] J.-H. Chang and L. Tassiulas. Energy conserving routing in wireless adhoc networks. In *Proceedings of the IEEE Conference on Computer Communications (Infocom)*, pages 22–31, March 2000.

[18] S. Singh, M. Woo, and C. Raghavendra. Power-aware routing in mobile ad hoc networks. In *Proc. Mobicom*, 1998.

[19] Marshall Hall Jr. *Combinatorial Theory.* Blaisdell, Waltham, Ma., 1967.

[20] Dimitri Bertsekas and Robert Gallager. *Data networks (2nd ed.).* Prentice-Hall, 1992.

[21] W. Shi, T. W. Sun, and R. D. Wesel. Quasiconvexity and optimal binary fusion for distributed detection with identical sensors in generalized gaussian noise. *IEEE Transactions on Information Theory*, 47:446–450, January 2001.

[22] P. F. Swaszek and P. Willett. Parley as an approach to distributed detection. *IEEE Trans. Aerospace and Electronic Systems*, 31:447–457, Jan. 1995.

[23] B. Hajek and G. Sasaki. Link scheduling in polynomial time. *IEEE Trans. Information Theory*, 34(5):910–917, September 1988.

[24] S. Ramanathan and E. L. Lloyd. Scheduling algorithms for multihop radio networks. *IEEE/ACM Trans. on Networks*, 1(2):166–177, April 1993.

[25] C. G. Prohazka. Decoupling link scheduling constraints in multihop packet radio networks. *IEEE Trans. on Computers*, 38(3):455–458, March 1989.

[26] R. L. Cruz and A. V. Santhanam. Optimal routing, link scheduling and power control in multi-hop wireless networks. In *Proc. Infocom*, 2003.

[27] A. Ephremides and T. V. Truong. Scheduling broadcasts in multihop radio networks. *IEEE Transactions on Communications*, 38(4):456–460, April 1990.

[28] S. A. Grandhi, R. Vijayan, and D. J. Goodman. Distributed power control in cellular radio systems. *IEEE Transactions on Communications*, 42:226–228, February 1994.

[29] G. J. Foschini and Z. Miljanic. A simple distributed autonomous power control algorithm and its convergence. *IEEE Trans. on Veh. Techn.*, 42(4):641–646, November 1993.

[30] S. Grandhi, J. Zander, and R. Yates. Constrained power control. *Int'l J. Wireless Personal Communications*, 1(4), 1995.

[31] H. Minc. *Nonnegative Matrices.* Wiley Interscience, 1988.

[32] R. A. Horn and C. R. Johnson. *Matrix Analysis.* Cambridge University Press, 1991.

[33] H. Karloff. *Linear Programming.* Springer Verlag, 1991.

[34] M. Grötschel, L. Lovasz, and A. Schrijver. The ellipsoid method and its consequences in combinatorial optimization. *Combinatorica*, 1(2):169–197, 1981.

[35] R. Sinkhorn. A relationship between arbitrary positive matrices and doubly stochastic matrices. *Ann. Math. Stat.*, 35:876–879, 1964.

[36] E. Lawler. *Combinatorial Optimization: Networks and Matroids.* Holt, Rinehart and Winston, 1976.

[37] K. Jansen and L. Porkolab. Preemptive scheduling on dedicated processors: applications of fractional graph coloring. In *Proc. Mathematical Foundations of Computer Science*, 2000.

[38] E. Scheinerman and D. Ullman. *Fractional Graph Theory.* Wiley, 1997.

[39] J. Zander. Distributed cochannel interference control in cellular radio systems. *IEEE Trans. Vehicular Technology*, 41(3):305–311, Aug. 1992.

[40] A. Berman and R. Plemmons. *Nonnegative Matrices in the Mathematical Sciences.* Academic Press, 1979.

[41] D. J. Baker and A. Ephremides. The architectural organization of a mobile radio network via a distributed algorithm. *IEEE Transactions on Communications*, COM-29(11):1694–1701, Nov. 1981.

[42] K. Nakano and S. Olariu. Energy-efficient initialization protocols for radio networks with no collision detection. *IEEE Trans. Parallel and Distrib. Sys.*, 11(8), 2000.

[43] S. Verdu. *Multiuser Detection*. Cambridge University Press, 1998.

[44] R. Zheng, J. Hou, and L. Sha. Asynchronous wakeup for ad hoc networks. In *Proceedings of ACM MobiHoc*, 2003.