PAMAS – Power Aware Multi-Access protocol with Signalling for Ad Hoc Networks*

Suresh Singh
Deartment of Electrical & Computer Engineering
Oregon State University
Corvallis, OR 97330
email: singh@ece.orst.edu

C.S. Raghavendra
Aerospace Corporation
El Segundo, CA 90245
email: raghu@aero.org

Abstract

In this paper we develop a new multiaccess protocol for ad hoc radio networks. The protocol is based on the original MACA protocol with the adition of a separate signalling channel. The unique feature of our protocol is that it conserves battery power at nodes by intelligently powering off nodes that are not actively transmitting or receiving packets. The manner in which nodes power themselves off does not influence the delay or throughput characteristics of our protocol. We illustrate the power conserving behavior of PAMAS via extensive simulations performed over ad hoc networks containing 10–20 nodes. Our results indicate that power savings of between 10% and 70% are attainable in most systems. Finally, we discuss how the idea of power awareness can be built into other multiaccess protocols as well.

1 Introduction

Ad Hoc networks are multi-hop wireless networks where all nodes cooperatively maintain network connectivity. These type of networks are useful in any situation where temporary network connectivity is needed. For instance, consider the problem of establishing a temporary wireless network in a region hit by some natural disaster. An ad hoc network here would enable medics in the field to retrieve patient history from hospital databases (assuming that one or more of the nodes of the ad hoc network are connected to the Internet) or allow insurance companies to file claims from the field. Other examples of such *ad hoc* networks include internetworking participants in a meeting to enable them to exchange data, battlesite networks, etc.

Nodes in an ad hoc network communicate via radio and since the radio channel is shared by all nodes, it becomes necessary to control access to this shared media. Several authors have developed channel access protocols for multi-hop radio networks where the goal has been maximizing throughput and minimizing transmission delay. Unlike this previous work, however, in this paper we present a channel access protocol that reduces the **power consumption** at each of the nodes. Reducing power consumption is clearly an important goal because battery life is not expected to increase significantly in the coming years. In an ad hoc network, it is even more important to reduce power consumption because these networks are typically established in mission critical environments (such as disaster relief).

Significant power is consumed at a node when it either transmits a packet or when it receives a packet. Thus, the DEC Roamabout radio [3] consumes approximately 5.76 watts during transmission, 2.88 watts during reception and 0.35 watts when idle. The radio used in [13] consumes 15 watts while transmitting, 11 watts while receiving and 50mW in idle mode. Now, notice that in ad hoc networks, a transmission from one node to another is potentially overheard by all the neighbors of the transmitting node – *thus all of these nodes consume power even though the packet transmission was not directed to them*! For example, in the ad hoc network illustrated in Figure 1, node A's transmission to node B is overheard by node C because C is a neighbor of A. Node C thus expends power receiving a packet not sent to it! In our protocol, node C *turns itself off* during the transmission from A to B to conserve power. It is easy to see that the potential savings of this simple approach can be enormous – particularly in dense networks.

*The first author's work was supported by the NSF under grant number NCR-9706080 and ONR under grant number N00014-97-1-0806.

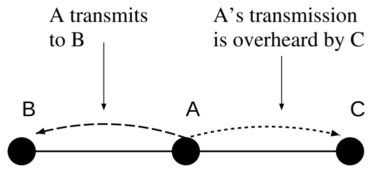

Figure 1: Unnecessary power consumption.

The remainder of this paper is organized as follows. In the next section we survey several multiaccess protocols that have been devised for ad hoc networks. Section 2.1 presents other work related to conserving power. Our protocol is presented in section 3 and its performance is discussed in section 4. Section 5 discusses how our approach for conserving power can be extended to other multiaccess protocols. We also discuss possible extensions of our basic protocol. Finally, our conclusions are presented in section 6.

2 Channel Access Protocols for Ad Hoc Networks

Channel access protocols for ad hoc networks have to contend with the problem of *hidden terminals* in addition to the problem of contention as in the Ethernet. Figure 2 illustrates the hidden terminal problem. Here, node A begins transmitting a packet to node B. However, since node C is out of range of node A, it begins transmitting a packet some time later. This results in collisions at the *receiver* B. Observe that neither of the two senders is aware of the collision and therefore cannot take preventive measures. It is noteworthy that this type of a problem does not arise in the Ethernet (for instance) because all nodes can hear one another. Several authors have developed different solutions to the hidden terminal problem. In this section we examine some of these proposals. Before doing so, however, it is important to observe that research in building ad hoc packet radio networks was initiated by DARPA in 1972. An excellent summary of the results of this work are provided in [18, 4]. Many access protocols developed as part of this program used some form of CSMA. Suggested approaches for dealing with hidden terminals include using appropriate "randomization delaying" [16] to reduce the probability of receiver-side collisions, the use of CDMA [11] and the use of adaptive transmission scheduling [9] (based on node connectivity) which ensures that the probability of two nodes transmitting to a common receiver is small. [27] presents another spread-spectrum based access protocol called PSMA (Preamble Sense Multiple-Access) that is a variant of CSMA where nodes wait for a random time before transmitting. A brief description of some of the more recent radios developed for military applications can be found in [2]. In the remainder of this section we will examine multiaccess protocols that PAMAS is based on. We will focus on how these protocols deal with (or not) the hidden terminal problem.

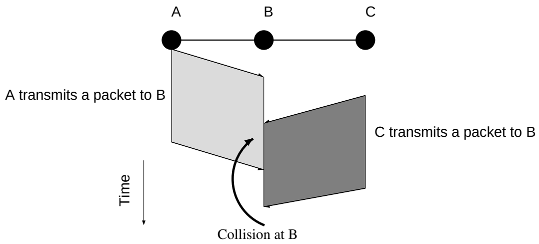

Figure 2: The hidden terminal problem.

MACA[17] is a protocol that many other, more recent, protocols are based on. Here, whenever a node wishes to transmit a packet to a neighbor, it first transmits a RTS (Request To Send) message. The receiver responds with a CTS (Clear To Send) message. Upon receiving the CTS message, the sender begins transmitting the packet. How does this RTS-CTS message exchange alleviate the hidden terminal problem? In Figure 2, C would have received the CTS transmission from B before A begins transmitting the packet to B. Thus C can hold off transmitting until B receives A's packet completely (the RTS and CTS messages contain the length of the packet). It is possible that the RTS message or its CTS may suffer a collision. In this case the sender executes a binary exponential backoff algorithm and tries to send a RTS again, later.

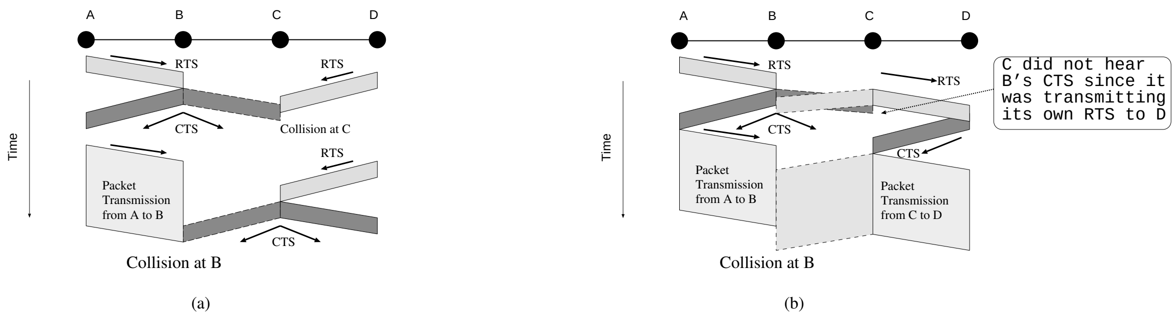

Figure 3: Hidden terminal problem in MACA.

Even though MACA solves the hidden terminal problem illustrated earlier, it creates another! In Figure 3(a), A sends a RTS to B who responds with a CTS. However, D sends a RTS to C at about the time that B is sending a CTS to A. Thus C hears a collision and does nothing. A receives the CTS and begins transmitting the data packet. D, on the other hand, retransmits a RTS after some time. Since C does not know that B is receiving a packet, it responds with a CTS. This CTS transmission, unfortunately, collides with the packet transmission at node B.

When collisions occur in MACA, recovery is left up to the transport layer thus greatly reducing throughput. MACAW[5] is a modified version of MACA where link layer ACKs have been added. Thus, a sender can retransmit a packet that was not successfully received at the receiver. In the above example, B will not send an ACK for the packet and A will retransmit it again. FAMA[12] is a refinement of the MACAW protocol in that it includes a non-persistent CSMA at the beginning of each free slot. In addition, the length of the CTS is made longer than the RTS to deal with the situation illustrated in Figure 3(b). If the length of the CTS is longer, node C will receive a part of the CTS transmission (either the initial part or the end). It will interpret this as noise wait for the length of one (maximum length) packet transmission before doing anything (it will not transmit a packet even if it receives a CTS from D). This solution also fixes the problem illustrated in Figure 3(a). Here, node C will hear noise (when control packets collide) and will ignore all transmissions for the length of time taken to transmit one maximum length packet. MACA/PR[20] is a protocol based on MACAW with the provision of non-persistent CSMA (as in FAMA). In addition, MACA/PR supports real-time data traffic by including a reservation mechanism in the RTS-CTS-Packet-ACK sequence. Finally, the IEEE 802.11 standard for wireless LANs includes the collision avoidance of MACA and MACAW. In addition, all directed traffic uses positive ACKs (again as in MACAW).

A different approach to the problem is described in [29]. In this system, there is a separate channel used for transmitting "busy tones". Thus, when a node wants to transmit a packet, it transmits the preamble of its packet (this contains the receivers address). The receiver responds with a busy tone. On hearing the busy tone, the sender continues sending the packet. It is easy to see that the hidden terminal problems described above are handled with this solution. Other approaches to channel access include splitting the network into clusters and using a different spreading code in each cluster (see [14]).

2.1 Power Conserving Research

In this section we review work related to conserving power in devices used in mobile environments. Battery life imposes a severe constraint on the deployment and large-scale use of mobile computing technology in the future and has prompted several researchers to develop approaches for conserving power on mobile computers. Significant amount of power is consumed by the display, by spinning disks, by the CPU and by the radio. In the event a camera or other recording device is attached to the mobile that I/O device is also likely to be a power drain. [15] discusses the problem of power consumption in displays and proposes several solutions. [10, 19, 36] study the issue of power consumption by spinning disks and propose algorithms for spinning up or spinning down disks based on the expected disk access patterns. [7] is one example of research on the development of power-efficient I/O devices – in this case the design of a power-efficient wireless digital camera suitable for use in the field. [23] presents a useful discussion of energy consumption in a prototype multimedia radio built at UCLA.

Recently, some researchers have begun studying the problem of reducing power consumption by the wireless interface. Thus, [32] observes that the average life of batteries in an *idle* cellular phone is one day. [33] studies power consumption of several commercial radios (WaveLAN, Metricom and IR) and observes that even in Sleep mode the power consumption ranged between 150-170 mW while in Idle state the power consumption went up by one order of magnitude. [32] observes that the only way to reduce power consumption in radios is to *shut them off* and use this observation to propose a power efficient MAC layer protocol suitable for use within one cell for communication between a base station and the mobiles. This protocol is based on the paging protocols POCSAG and FLEX where a base station periodically transmits a beacon followed by a minislot containing the ID of nodes that have a page waiting for them. These nodes remain awake in order to receive their messages while all the others power themselves off. A similar idea (based on reservation) is included in the IEEE 802.11 standard as well (see [26]). Here, nodes transmit their requests to the base station during specific reservation intervals and the base station transmits a TIM (Traffic Indication Map) that includes the transmission schedule for the nodes. All nodes not participating in transmission or reception of packets go into doze mode until the next reservation period. The standard also includes an extension of this idea to ad hoc single-hop networks. Here, nodes compete to be elected the leader to play the role of the base station. [30] presents a comparison of the power consumption behavior of three protocols – IEEE 802.11, DQRUMA (see [21]) and DSA++ (see [25]) – in a single-hop environment. Their main conclusions are that contention results in higher energy consumption while reservation and scheduling results in lower energy consumption. [8] also discusses the energy consumption of protocols and shows that persistence is not always a good choice and adaptive strategies that avoid packet retransmissions during bad channel periods is a good energy conserving strategy. Furthermore, [8] presents a access protocol for cellular networks based on ALOHA and reservation (the protocol is similar to IEEE 802.11) and analyze its performance (energy consumed and throughput). [31] also presents a reservation-based power conserving access protocol for mobile ATM networks.

It is interesting to note that all the work reported above has been done in the context of a cellular network model (i.e., all mobiles are one hop away from a base station). Our work, on the other hand, considers energy conservation in a multi-hop wireless network.

3 The PAMAS Protocol

The PAMAS protocol is a combination of the original MACA protocol (see [17]) and the idea of using a separate *signalling channel* (as in [29, 34]). Thus, we assume that the RTS-CTS message exchange takes place over a signalling channel that is separate from the channel used for packet transmissions. This separate signalling channel enables nodes to determine when and for how long they can power themselves off (section 5 shows why the lack of a separate signalling channel, in most other protocols, results in sub-optimal power conservation). In this section we first present the PAMAS protocol. Later, we add power conserving behavior to the protocol in a way that does not change the delay or throughput behavior of PAMAS.

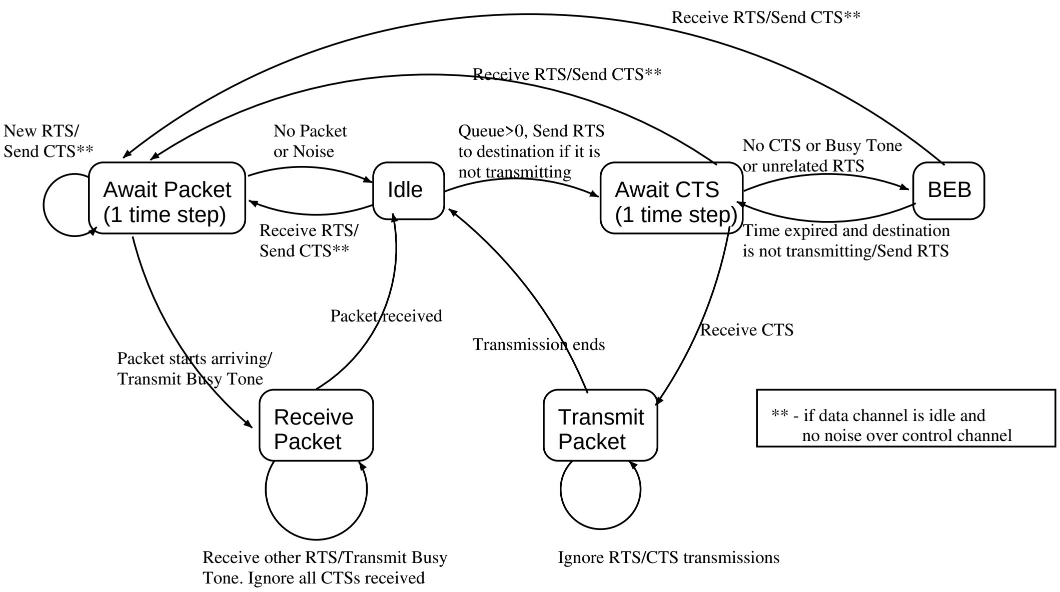

Figure 4: The PAMAS Protocol.

The state diagram outlining the behavior of our protocol is illustrated in Figure 4. As indicated in the figure, a node may be in any one of six states – *Idle, AwaitCTS, BEB* (Binary Exponential Backoff), *Await Packet, Receive Packet,* and *Transmit Packet.* When a node is not transmitting or receiving a packet, or does not have any packets to transmit, or does have packets to transmit but cannot transmit (because a neighbor is receiving a transmission) it is in the Idle state. When it gets a packet to transmit, it transmits a RTS and enters the AwaitCTS state. If the awaited CTS does not arrive, the node goes into binary exponential backoff (the BEB state in the figure). If a CTS does arrive, it begins transmitting the packet and enters the Transmit Packet state. The intended receiver, upon transmitting the CTS, enters the Await Packet state. If the packet does not begin arriving within one roundtrip time (plus processing time), it returns to the Idle state. If the packet does begin arriving, it transmits a busy tone over the signalling channel and enters the Receive Packet state. Let us now look at the functioning of the protocol in some more detail[1].

When a node in the Idle state receives a RTS, it responds with a CTS if no neighbor is in the Transmit Packet state or in the AwaitCTS state. It is easy for a node to determine if any neighbor is in the Transmit Packet state (by sensing the data channel). However, it is not always possible for a node to know if a neighbor is in the AwaitCTS state (the transmission of the RTS by that neighbor may have collided with another transmission over the control channel). In our protocol, if the node heard noise over the control channel within τ [2] of the arrival of the RTS, it does not respond with a CTS. If, however, it does not hear a packet transmission begin within the next τ, it assumes that none of its neighbors is in the AwaitCTS state anymore.

Now consider a node that is in the Idle state and has a packet to transmit. It transmits an RTS and enters the AwaitCTS state. If, however, a neighbor is receiving a packet that neighbor responds with a busy tone (twice as long as a RTS/CTS) that will collide with the reception of the CTS. This will force the node to enter the BEB state and not transmit a packet. If no neighbor transmits a busy tone and the CTS arrives correctly, transmission begins and the node enters the Transmit Packet state.

Say a node that transmitted a RTS does not receive a CTS message. It enters the BEB state and waits to retransmit a RTS. If, however, some other neighbor transmits a RTS to this node, it leaves the BEB state, transmits a CTS (if no neighbor is transmitting a packet or is in the AwaitCTS state) and enters the Await Packet state (i.e., it waits for a packet to arrive). When the packet begins arriving, it enters the Receive Packet state. If it does not hear the packet in the expected time (i.e., round trip time to the transmitter plus some small processing delay at the receiver), it goes back to the Idle state.

When a node begins receiving a packet, it enters the Receive Packet state and immediately transmits a busy tone (whose length is greater than twice the length of a CTS). If the node hears a RTS transmission (directed to some other node) or noise over the control channel at any time during the period that it is receiving a packet, it transmits a busy tone. This ensures that the neighbor transmitting the RTS will not receive the expected CTS. Thus, the neighbors transmission (which would have interfered with the node receiving a packet) is blocked.

It is easy to see that our protocol handles the hidden terminal problems illustrated in Figure 2. For instance, in the first example, node B's reception of a packet from A will not be affected by the transmission of a CTS by node C (since these transmissions occur over separate channels). In the second example, when node B begins receiving the packet from A, it transmits a busy tone that is heard by node C. If the busy tone overlaps with the CTS transmission from node D to node C, node C hears only noise and will enter the BEB state and transmit a RTS again, later. This retransmission of the RTS will be met by another busy tone from B if B is still receiving the packet. This continues until either B finishes receiving or D sends a RTS to C (in this case C may begin receiving a packet from D).

3.1 Powering off radios

We noted in section 1 that nodes consume power while transmitting or even while receiving a packet. Unfortunately, in an ad hoc network, it is frequently the case that a packet transmission from one node to another will be overheard by all the neighbors of the transmitter. All of these nodes will thus consume power needlessly. Consider a simple example where the network is fully connected (i.e., all nodes are within transmission range of each other) with n nodes. A transmission here will be heard by all $n-1$ nodes. If the power consumed in transmitting a packet is t and r is the power consumed while receiving, we see that the total power consumed (system-wide) for one packet transmission is $t+(n-1)r$. This is a huge waste because the total power consumed for a single transmission should be no more than $t+r$ (ignoring the power consumed in the CTS-RTS-Busy Tone transmissions).

[1] We assume that the data channel and the control channel have identical conditions (e.g., noise).

[2] τ = one roundtrip time plus transmission time for a RTS/CTS.

In order to conserve power and extend the lifetime of mobile nodes, the PAMAS protocol requires nodes to shut themselves off if they are in a situation where they overhear transmissions. Thus, our protocol ensures that in the fully connected example above, $n-2$ nodes will shut themselves off for the duration of the transmission. We have identified two conditions under which it is beneficial for a node to turn itself off.

- $\oplus$ If a node has no packets to transmit, then that node ought to power itself off if a neighbor begins transmitting.

- $\oplus$ Similarly, if at least one neighbor of a node is transmitting and another is receiving, the node ought to power off because it cannot transmit or receive a packet (even if its transmit queue is non-empty).

Every node in our system makes the decision to power off independently. A node knows if a neighbor is transmitting because it can hear the transmission (over the data channel). Likewise, a node (with a non-empty transmit queue) knows if one or more of its neighbors is receiving because the receivers transmit a busy tone when they begin receiving a packet (and in response to RTS transmissions). Thus, a node can easily decide when to power off. There are, however, two additional questions to be answered:

1. For how long is a node powered off?
2. What happens if a neighbor wishes to transmit a packet to a node that has powered itself off?

Let us answer the second question first using an example. Say we have a line network with four nodes (A–B–C–D) and node B is transmitting to node A. The transmission is overheard by node C (who powers itself off). Say node D has a packet to transmit to node C. Since C is powered off, D's RTSs go unanswered causing D to go into BEB. What happens if C was not powered off? In this case, since C's neighbor B is transmitting a packet, C will not respond to D's RTSs. Thus, C's behavior, from the viewpoint of D, is the same irrespective of whether C is powered off or not! As a corollary, we can see that packet delays *do not increase* as a result of powering off nodes. This is because the period of time when a node is powered off is one where it can neither receive packets nor can it transmit packets.

To answer the first question we need to consider several cases. Ideally, a node ought to stay powered off whenever any of the two conditions ($\oplus$) hold. However, collisions in the signalling channel and the data channel may make it difficult for a node to determine the length of a transmission. Nodes follow the following protocol to determine the length of time for which they can power off.

- $\odot$ When a packet transmission begins in the neighborhood of a node, it knows the duration of that transmission (say l). If the node has an empty transmit queue, it powers itself off for l seconds.

- $\odot$ It is possible that one or more neighbors may begin data transmission while the node is powered off. In this case, when the node powers back on, it will continue hearing transmissions over the data channel. If the node still has an empty transmit queue, it ought to power itself off again. But for how long[3]?

  Figure 5 illustrates the case when three neighbors begin transmission after a node powers off. These transmissions are ongoing when the node powers back on and it needs to find out the remaining transmission time (i.e., it needs to find out the value of l_2). To do this we add additional functionality in our protocol. The node, upon waking up, transmits a *t_probe(l)* packet over the control channel where l is the maximum packet length. All transmitters with transmissions completing in time period $[l/2, l]$ respond with a *t_probe_response(t)* packet (t is the time when this transmitter's transmissions will end). If the packet is received without collision, the node powers itself off until time t. Otherwise, if there was a collision, it probes interval $[3l/4, l]$. If there is silence, it probes $[l/2, 3l/4]$, and so on. If there was silence in response to the initial probe, it probes interval $[0, l/2]$. In effect, the node does a *binary search* to determine the time when the last (current) transmission will end.

  A simplification that can be built into the probe protocol just described is the following. When the node hears a collision in response to a probe of the interval $[t_1, t_2]$, it powers itself off for the period t_1. This simplification attempts to reduce the probing time by sacrificing some battery power (since the node will power back on while a transmission is ongoing). Observe that if the node powers itself off until time t_2 (instead of t_1), there is a likelihood that packet delays will increase. This is because the packet transmissions may cease soon after time t_1 but the node is powered off until t_2. Thus if another node has a packet for this (powered off) node, that packet cannot be delivered.

[3]The node powers off its control channel as well and therefore does not know the length of the remaining transmissions

⊙ Next consider the case when a node has a non-empty transmit queue. When the node powers back on (after it first powered off), it transmits a RTS (rather than probing, because it needs to transmit a packet). If any node in its neighborhood is receiving a transmission, that node responds with a busy tone (containing the length of the remaining transmission). If the busy tone collides with another busy tone or a CTS or some other RTS, the node attempts to probe the *receivers* using the same binary search algorithm described above but using a *r_probe(l)* packet (the prefix r denotes a receiver probe packet). It probes the transmitters next using the *t_probe(l)* packet. Then it powers itself off for $min\{r,t\}$ where r is the time the last receiver finishes receiving and t is the time the last transmitter finishes transmitting.

To understand the reason for taking a min above, consider the following two cases. If $t < r$ (i.e., all transmitters finish before the receivers finish) then we need to power on this node so that, if some other node has a packet for this node, that node can go ahead with its transmission (to reduce delays). If, on the other hand, $t > r$, we need to power back this node after r so that it can begin transmitting packets from its queue (again to keep delays small).

⊙ Finally, it is important to observe that the probe messages could get corrupted (say more than one node powers on at the same time and transmits a probe message). In this case there will be no response to the probes and the nodes will stay on. A workaround we suggest (but have not implemented) is to use p-persistent CSMA when transmitting a probe packet (with p choosen appropriately). This will reduce the possibility of collisions of probe packets. We have not implemented this scheme because of two reasons:

- Under light load conditions, the probability of hearing a new transmission after a node powers back on is low. Hence there is no need to build a sophisticated protocol.
- Under heavy loads, it is almost always the case that when a node powers on there will be ongoing transmissions. In this case, it is unlikely that the node will have an empty transmit queue. So it will try to transmit RTS messages which will evoke busy tone responses from receivers. This will quickly inform the node of the additional time it needs to power off for (we assume that the busy tone transmissions include the length of the remaining transmission).

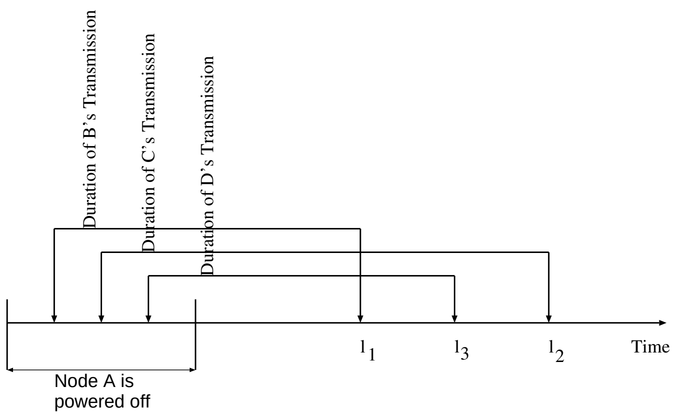

Figure 5: Situation when a node powers back on.

It is noteworthy that the above *probe protocol* can be simplified considerably if we assume that the node only powers off its data interface but always leaves the signalling interface powered on. This will enable the node to always know the length of new transmissions and keep the data interface powered off appropriately. Figure 6 illustrates the block diagram needed for this type of a communications device. Here, the signalling interface listens to all RTS/CTS/Busy Tone transmissions and records the length of each transmission and reception. This information (along with the length of the transmit queue) is fed to the power aware logic which determines whether to turn the data interface off or on. The power aware logic that can be used is as described above with the exception of the probe algorithm.

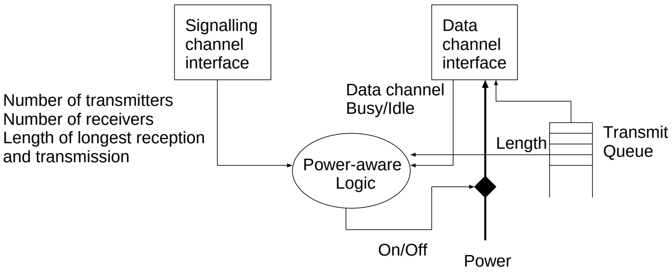

Figure 6: Separate interfaces for signalling and data.

3.1.1 Effect of powering off on Delay and Throughput

An important concern related to powering off radios is whether the delay or throughput behavior of PAMAS changes. We claim that powering off radios, as we have described, does not have any effect because a radio powers itself off if it either cannot receive data transmissions directed to it (because a neighbor is transmitting a packet) or if it cannot transmit a packet (because a neighbor is receiving another transmission). In these cases, even if the radio was not turned off, it could not receive/transmit a packet. Thus, powering off the radio has no effect on the behavior of PAMAS. This statement does have an important caveat, however. The *length* of time that a radio is powered off should be no longer than necessary (i.e., the two conditions mentioned above hold for this time period) otherwise the delay and throughput behavior of PAMAS will be changed. It is for this reason that in section 3.1 we use the probe algorithm to enable nodes to estimate the length of time that a radio can turn itself off. In fact, as we noted in the discussion earlier, we err on the side of caution and may underestimate the length of this period. This results in sub-optimal power savings but ensures that the delay/throughput behavior of PAMAS remains unchanged.

4 Power Conserving Behavior of PAMAS

In order to characterize the power (or energy) conserving behavior of our protocol we conducted extensive simulations where we compared the energy expended by PAMAS without power conservation and PAMAS with power conservation. The simulations were conducted in three different network topologies that, we believe, represent most ad hoc networks. Thus, we ran PAMAS in a *random network* topology, a *line topology* and a *fully connected network* topology. We conjectured that networks where nodes are densely connected will show the most power savings while networks that are sparse will show the least amount of power savings. The reason is that, in a dense network, if one node transmits most of its neighbors can power off. In a sparse network, on the other hand, fewer nodes can power off because more simultaneous transmissions are possible. An implication of this is that the throughput will typically be higher in sparse networks because more transmissions can go on simultaneously.

In the simulations, we used fixed size packets (512 bytes). The RTS and CTS packets were 32 bytes each and the busy tone was twice as long. The bandwidth was assumed to be 12.8Kbps (observe that our results also hold for higher data rates – we used this rate to keep the length of the simulation time small). In terms of power conservation, we assumed that no power is consumed when a node is idle (i.e., powered on but not hearing any transmissions), 1 unit of energy is consumed for 32 bytes of transmission (over either the data channel or the signalling channel), and, 0.5 units of energy are consumed at a receiver for every 32 bytes processed (again, data or control).

We ran simulations for networks with 10 and 20 nodes to see how energy conservation scaled. For the random networks, we generated edges randomly uniformly with probabilities between 0.1 to 0.9 for different experiments (we only used connected networks in our experiments). A probability of 0.1 generates sparse networks (allowing more parallel transmissions and hence less energy savings) while an edge probability of 0.9 yields dense networks with much better energy conserving behavior. Traffic arrived at *each* node according to a poisson process. The destination was choosen randomly uniformly from the remaining $n-1$ nodes and the packet was routed using the shortest path. All nodes maintain a FIFO buffer of packets awaiting transmission (we call this the Transmit Queue). The length of this buffer is fixed at $2n$ per nodes. Packets arriving at a node with a full buffer are dropped.

Finally, to measure the power savings, we calculated the total number of bytes transmitted B_t during a run of the experiment (this included data bytes as well as control bytes), the total number of bytes received B_r(note that a packet may be received by more than one node and is therefore counted more than once) and the total number of packets P transmitted during the experiment. The energy expended per packet is then,

$$\mathcal{P} = (B_t + 0.5 \times B_r)/P$$

If we use power conservation, the number of bytes received tends to be smaller, say it is B_r^c. Then, the energy expended is,

$$\mathcal{P}^c = (B_t + 0.5 \times B_r^c)/P$$

and the power *savings* can be written as,

$$\text{Percentage of Power Saved } = (\mathcal{P} - \mathcal{P}^c)/\mathcal{P}$$

We ran each experiment 150 times and computed 95% confidence intervals for the percentage of power saved. The interval half-widths kept to less than 5% of the point values in all cases.

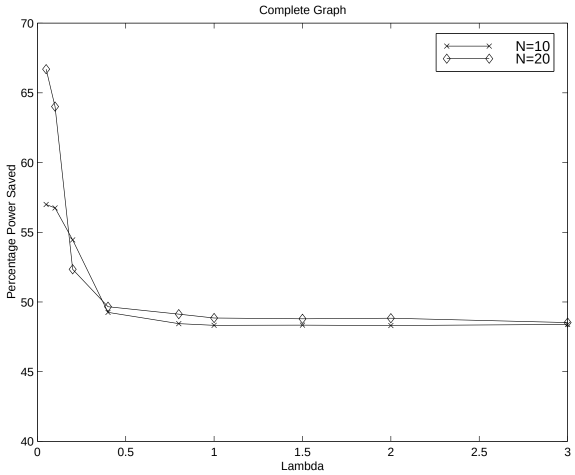

Figure 7: Power saved in complete networks with 10 or 20 nodes.

Figure 7 plots the power saved in a *complete network* (i.e., fully connected) topology as a function of load λ (packets/sec/node). We ran the experiments with 10 nodes and with 20 nodes. It is easy to see that PAMAS reduces power consumption by almost 50% for high loads and by even more for low loads. The reason for the higher savings at low loads is that, at low loads, there is less contention for the channel resulting in fewer control message transmissions (i.e., RTS/CTS/Busy Tone transmissions). At high loads, almost all the nodes have packets to send and thus contention for the channel is high. This results in fewer actual packet transmissions (which is the only time when $n-2$ of the n nodes not involved in the transmission can power off) and lower power savings. Finally, observe that the power savings at low loads are higher for networks with more nodes. This is because the number of nodes that can power off is greater when there are more nodes in the network.

The complete network case illustrates the best case performance of PAMAS. At the opposite end of the spectrum we have a *line network* where, as expected, the savings in power were not as dramatic, see Figure 8. The savings here range from 20% at light loads to less than 10% at heavy loads. The reason for these lower savings is that in a line network, a large number of packet transmissions can go on simultaneously. Thus, fewer nodes are in a position to overhear unintended transmissions.

Finally, to get an idea of the dependence of power savings on node connectivity, we ran simulations with random networks. Figure 9 illustrates the power saved as a function of edge probability for $\lambda = 0.05, 0.1, 0.5, 1.0$ and 4.0 pkts/sec/node in ten node networks. Figure 10 plots the same values for a network with 20 nodes. As these figures show, power savings increase as the network connectivity increases. This is not surprising because in dense networks, more nodes can power off during transmissions. In sparse networks, at light load, we obtain 20-30% power savings while at high loads this drops to 10%. In dense networks, the power savings at light loads is 60-70% while at high loads this drops to 30-40%. This drop is caused because the length of the *contention period* increases.

In the following subsection we develop approximate bounds on the optimal energy savings that can be achieved in the different types of networks we have studied. The mathematical formulation of these bounds illustrates the reasons for the high degree of energy savings PAMAS achieves.

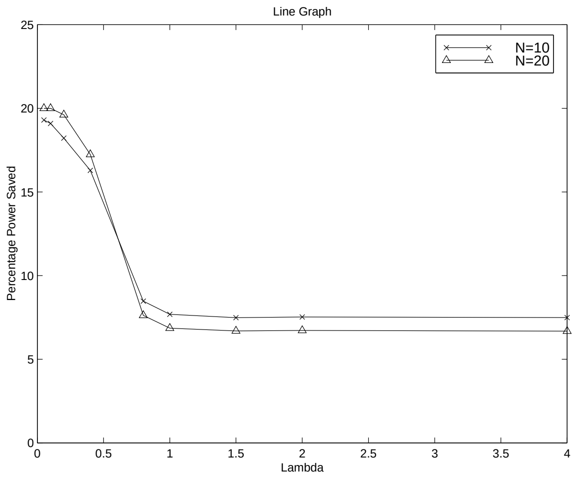

Figure 8: Power saved in line networks with 10 or 20 nodes.

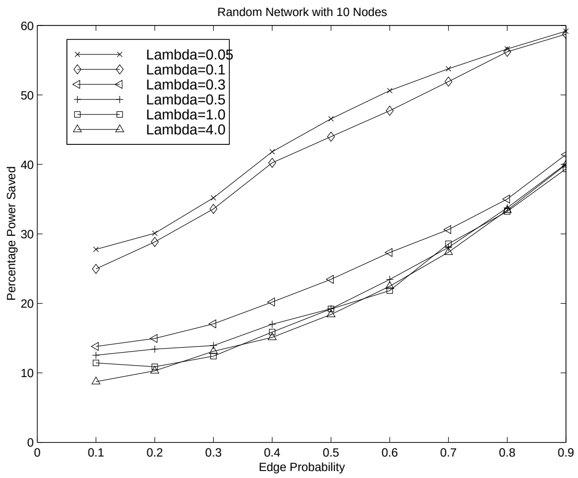

Figure 9: Power saved in random networks with 10 nodes.

4.1 Bounds and Approximations on Energy Savings

An ad hoc network can be modeled by a graph, where the nodes represent the mobile radio units and edges represent neighboring nodes. Battery power is consumed by a radio when transmitting or receiving a packet. Most radios available in the market operate by consuming about twice as much power (P_t) when in transmit mode compared to being in receive mode (P_r). A radio also consumes a very small amount of power when in idle mode (P_i), i.e., powered on, but neither transmitting nor receiving. Whenever a node k is powered on and a neighbor node transmits a packet, it will consume power P_r even if the transmission is not intended for k. Battery power can be saved if we can turn off the radios whenever a neighbor transmits packets not intended for that node. For this analysis, we consider normalized power consumption by a node to be 1 unit in transmit mode, 0.5 unit in receive mode, and 0 units in idle mode.

Assuming point-to-point communication, we can establish some bounds on power savings in an ideal situation. When a radio transmits a packet, it will be intended for one of its neighbors. Therefore, ideally, for each packet transmission exactly one intended neighbor should be powered up to receive the packet and the rest of the nodes should be powered off to maximize power savings. In a fully connected topology of n nodes, for each packet transmission, $n-2$ nodes can be powered off. Our PAMAS protocol achieves this maximum power savings as the unintended nodes can power themselves off during each packet transmission. In addition, all the nodes know exactly how long to turn themselves off, and therefore, our protocol is optimal for the fully connected topology.

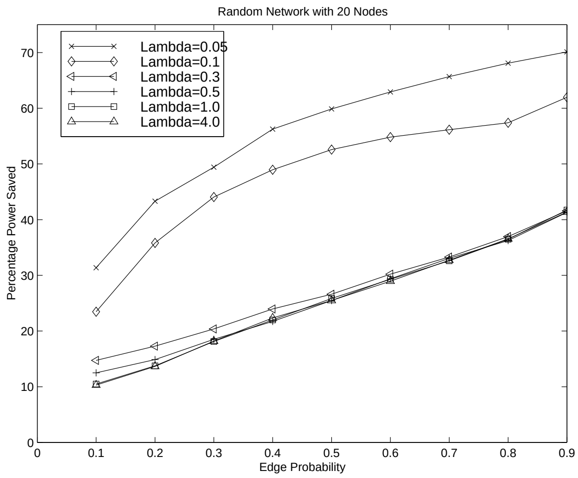

Figure 10: Power saved in random networks with 20 nodes.

In the remainder of this section we develop *bounds* on the maximum possible energy savings for different network models. In section 4.2 we derive the bounds for the case when the network is a line. Section 4.3 presents bounds for the case when we have random network topologies (as in our simulations). Finally, in section 4.4 we develop bounds on energy savings over the set of *all random networks*. These bounds are useful when we consider that, in a real life situation, nodes are mobile and, as a result, the network topology changes frequently. Thus, an average bound over all network topologies gives us an idea of the life of such mobile ad hoc networks.

4.2 Bounds for the Line Topology

Consider a line topology with n nodes. Each node has at most two neighbors. In this topology, when a middle node is transmitting one of its neighbors should be turned off if that neighbor is neither transmitting nor able to receive without interference. In the Figure below, we show various scenarios with transmitters and receivers being paired off. A node with symbol T indicates it is a transmitting node, a node with symbol R indicates a receiving node, and a node with symbol O can be off. Note that two adjacent nodes can both be transmitting to their neighbors on the opposite sides without interference. Figure (a) shows the most tight packing where almost all nodes are transmitting or receiving. Figures (c,d) show the situation where the most energy savings is possible. Here, for every transmitting node one neighbor can be turned off as it is neither transmitting nor receiving. Figure (b) shows the situation which is somewhere in between. In the following, we will derive the bounds on energy savings by considering light load and heavy load conditions.

(a) T—-R—-R—-T—-T—-R—-R—-T—-T—-
(b) T—-R—-R—-T—-O—-T—-R—-R—-T—-
(c) O—-T—-R—-O—-T—-R—-O—-T—-R—-
(d) R—-T—-O—-O—-T—-R—-R—-T—-O—-

Theorem 4.1 *In a line topology, with uniform traffic, the amount of energy savings due to the PAMAS protocol for large n, under light load conditions, is 20%.*

Proof: In the line topology, each node that has a packet to transmit will have an intended receiver as one of its two neighbors. Under light load conditions, we assume that transmitters are not nearby. Specifically, a transmitting node will not have a competing neighbor trying to transmit. Let the nodes in the line topology be numbered from 1 through n from left to right. With uniform traffic, if the transmitting node is the kth node, its intended receiver will be its right neighbor with a probability of $(n-k)/(n-1)$ and left neighbor with a probability of $(k-1)/(n-1)$. Under light load conditions, the neighbor who is not the receiver can be turned off to save 0.5 units of energy.

Thus, the total energy savings with the PAMAS protocol for the kth node being the transmitter under light load conditions is

$$\frac{(n-k)}{(n-1)} \times 0.5 + \frac{(k-1)}{(n-1)} \times 0.5$$

which is 0.5 units of energy.

Under these load conditions, an upper bound on the total power savings is then $0.5/(1.5 + .5)$ without control overhead is 25%.

Now, we will add the power consumed in acquiring the channel. The transmitter sends an RTS packet (32 bytes) which will be received by the two neighbors. The intended receiver replies with a CTS which is received by two nodes. Finally, the receiver transmits a busy tone of length 64 bytes which is heard by its two neighbors. Total control overhead is equivalent to transmission energy of 8*32 bytes which is 0.5 units of energy. With this control overhead, the upper bound on the total power savings for this light load condition is $0.5/(1.5 + .5 + .5)$, which is 20%.

Thus, under light load conditions, the power savings due to PAMAS is bounded by 20%. □

Theorem 4.2 *In a line topology, with uniform traffic, the amount of energy savings due to the PAMAS protocol for large n, under heavy load conditions, is bounded by 15% without control overhead and by 12.5% with control overhead.*

Proof: When the load is heavy, we expect that every node has packets to transmit. Let us consider the situation with some middle node k being the transmitter. Its intended receiver will be one of its two neighbors. There are four possible cases with node k being the transmitting node depicted as the third node in the following figure. We can determine the probabilities for these four events under uniform traffic assumption. The probability that its intended receiver is node $k+1$ is given by $(n-k)/(n-1)$ and the probability that its intended receiver is node $k-1$ is given by $(k-1)/(n-1)$. With node $k+1$ selected as the receiver, we can have two cases where node $k-1$ can either successfully transmit to node $k-2$ or be off. The probability that node $k-1$ can successfully transmit to node $k-2$ is $(k-2)/(n-1)$ if its head of line packet has a destination lower than $k-1$. The probability that node $k-1$ will be off because it cannot transmit is $(n-k+1)/(n-1)$ when its head of line packet has a destination higher than $k-1$. Likewise, we can have two sub cases when the intended receiver for node k's transmission is node $k-1$. There will be power savings when one of the two neighbors can be turned off. There will be no energy savings with node 1 or node n being the transmitter.

(1) —-R—-T—-T—-R——-
(2) ———O—-T—-R——-
(3) ———R—-T—-T—-R–
(4) ———R—-T—-O——-

By varying the value of k from 2 to $n-1$ as being the transmitter, the total energy savings for the entire topology can be computed. Since the power savings is dependent on k, and hence the node position, to find the total power savings, we will consider only cases 2-4 above. This is to avoid double counting – we must include only the three cases (2-4 above) as case (1) of node k would be included in the calculation for node $k-1$ (in case (3)). Power savings with node k being the transmitter with this modification is then,

$$\frac{(n-k+1)}{(n-1)} \times 0.5 + \frac{(k-1)}{(n-1)} \times \left(\frac{(n-k-1)}{(n-1)} \times 0 + \frac{k}{(n-1)} \times 0.5\right)$$

Total power spent in these three cases without powering off radios is given by

$$\frac{(n-k+1)}{(n-1)} \times 2.0 + \frac{(k-1)}{(n-1)} \times \left(\frac{(n-k-1)}{(n-1)} \times 3.0 + \frac{k}{(n-1)} \times 2.0\right)$$

Percentage power savings due to PAMAS protocol with node k being the transmitter is given by,

$$PS(k) = \frac{((n-k)(n-k+1) + k(k-1)) * 0.5}{(k-1)(n-k-1) * 3.0 + ((n-1)(n-k+1) + k(k-1)) * 2.0}$$

By varying the value of k from 2 to $(n-1)$, we can find a bound for the average amount of power saved to total power consumed due to powering off of nodes and this value for heavy load condition is about 15%.

There is additional power consumed by the control messages and this needs to be estimated for each of the three cases above (2-4). A node's transmission of a RTS will collide with the RTS transmissions of one (or both) of its neighbors resulting in later retransmissions of the RTS message. The number of times a node will transmit a RTS message is thus at least 2. Therefore, it is reasonable to assume that this control overhead is twice as much as in the light load case. This overhead then amounts to equivalent transmission energy for 16*32 bytes or 1 unit of energy. When we account for this control overhead, the total power savings is reduced to 12.5% (this value was computed numerically for $n \geq 10$). □

4.3 Bounds and Approximations on Power Savings for Random Networks

Let us now develop bounds for the case when the ad hoc network is modelled as a random network. These topologies are characterized by two parameters – the number of nodes n and the probability of an edge between any pair of nodes, p. To characterize the maximum possible power savings, we consider the case when the network is *lightly loaded* and the case when the network is *heavily loaded*. In the light load case, a transmitting node will not find another transmitter nearby. Therefore, we can power off all its neighbors except the receiver. In the heavy load case, on the other hand, most nodes will have packets to transmit. Therefore, there will be several simultaneous transmissions and the only nodes that will power off will be those that (a) have a neighbor who is receiving a transmission, and (b) are not themselves receivers. Let us first derive the bounds for the light load case.

Theorem 4.3 *If the average number of neighbors of a transmitting node is d then the approximate amount of power savings (under light load conditions) is given by,*

$$\mathcal{S}_{light} = \frac{d-1}{1.25d + 2.5} \tag{1}$$

Proof:
Under light load conditions, we can assume that a transmitter will not find another transmitter nearby. Prior to transmitting a data packet, the transmitter and receiver exchange RTS/CTS packets. In addition, the receiver transmits a Busy Tone when it starts receiving the data packet. The length of the RTS/CTS packets is 32 bytes each while the Busy Tone is 64 bytes. Thus, 128 bytes are transmitted prior to the transmission of the 512 byte data packet.

The total energy expended in transmitting the control and data packets is $1+0.25$ units (we assume that it takes 1 unit transmit energy to transmit 512 bytes). In the case where we do not power off radios, the total energy expended in receiving the transmissions is $0.5 \times (1.25d)$ (recall that we are assuming that receiving a transmission consumes 0.5 units of power for every 512 bytes)[4]. The energy consumed when we do power off radios is $0.5 \times 0.25d + 0.5$ (the first term is the energy consumed by all neighbors in listening to the control packets and the second term is the power consumed by the intended receiver). Thus, the power savings can be written as,

$$\mathcal{S}_{\text{light}} = 1 - \frac{(1+0.25) + (0.5 \times 0.25d + 0.5)}{(1+0.25) + 0.5 \times (1.25d)} = \frac{d-1}{1.25d + 2.5}$$

This concludes the proof. □

The average degree of a node in a random graph with edge probability p can be approximated as $p(n-1)$ for $p > 0.3$. However, for smaller values of p, we need to do a more formal analysis. Specifically, we need to determine d *conditioned* on the fact that the graph is connected. This derivation is presented in Appendix A.

Let us consider the *heavy load* case next. In the heavy traffic load situation as many nodes as possible will be transmitting simultaneously without interference. For each transmitting node another node will be the intended receiver. Therefore, the number of simultaneous transmitters can vary from 1 to $n/2$ as we need an equal number of receivers. For each transmitter, an edge must exist to a receiving node and there cannot be any edges from that receiving node to the other transmitting nodes. Based on this observation, we have,

Theorem 4.4 *The average power savings (under heavy load conditions) in random graphs with an edge probability p is,*

$$\mathcal{S}_{Heavy} = \frac{n - 2\mu}{n + \mu(1.25 + 0.25d + 0.0625d\log d)} \tag{2}$$

where μ is the average value of the number of simultaneous transmitters in a random graph with an edge probability of p.

Proof: The energy consumed by the transmitters in transmitting the *data* packets is μ (since one 512 byte packet consumes 1.0 units of energy). The energy used, in the case when radios are not powered off, is then,

$$\mu + 0.5 \times (n - \mu) = 0.5n + 0.5\mu$$

[4]We assume that the random graph is symmetric (i.e., the expected degree of all nodes is the same), see [6].

The energy expended if we use PAMAS is,

$$\mu + 0.5\mu = 1.5\mu$$

and the power savings (without accounting for the control overhead) is,

$$1 - \frac{1.5\mu}{0.5n + 0.5\mu} = \frac{n - 2\mu}{n + \mu} \tag{3}$$

In both cases, some energy is used in the initial exchange of control messages. Unfortunately, unlike the light load case, RTS/CTS messages will collide due to the heavy load. We can estimate the number of RTS messages sent by one node as follows. If a node has d neighbors, in the first time step, all these nodes transmit a RTS. Since none succeed, in the next step, we assume that half of these nodes transmit a RTS, and so on. Thus, in $\log d$ steps, one node emerges the winner (note that this is a very optimistic scenario). The total number of RTS transmissions and the last CTS transmission is thus equal to $2d$. After a receiver starts receiving a data packet, it transmits a Busy Tone lasting twice as long as a RTS or CTS. Thus, the total energy expended in setting up one connection is,

$$e = 0.0625(2d + 2) + 0.5 * 0.0625 * \log d * d$$

where the first term represents the total transmit energy expended (recall that a RTS/CTS packet is 1/16th a data packet) and the second represents the total receive energy expended – the transmissions ($\log d$ in all) are heard by all d neighbors. The energy savings are,

$$\mathcal{S}_{\text{Heavy}} = 1 - \frac{1.5\mu + e\mu}{0.5n + 0.5\mu + e\mu} = \frac{n - 2\mu}{n + \mu + 2e\mu} = \frac{n - 2\mu}{n + \mu(1.25 + 0.25d + 0.0625d\log d)} \tag{4}$$

This concludes the proof. □

The value of μ can be estimated as follows. Say the maximum number of simultaneous transmitters in a network is k. Thus with k transmitters, we must have one edge from each of these k transmitters to some k receiving nodes and the remaining possible $k(k-1)$ from the transmitter set to receiver set must be absent as otherwise there will be interference. So far, we have accounted for $2k$ nodes as either transmitting nodes or receiving nodes. To make the remaining $(n-2k)$ nodes not be able to transmit, we require that each of these nodes have an edge to one of the k receivers. There are $k(n-2k)$ possible ways of choosing these additional $(n-2k)$ edges to ensure that we maximize the set of transmitters.

We can find the number of graphs with exactly k transmitters in a random graph with edge probability p to be,

$$\alpha = \binom{n}{k} \times \binom{(n-k)}{k} \times k! \times p^k \times (1-p)^{k(k-1)} \times k(n - 2 * k) \times p^{n-2k}$$

We can obtain the average value for the maximum number of transmitters that can simultaneously transmit in a random graph with edge probability p to be,

$$\mu = \frac{\sum_{k=1}^{n/2} k\alpha}{\sum_{k=1}^{n/2} \alpha}$$

We calculated the bounds for light load and heavy load cases for different values of n and p and the percentage of power savings varies from 10% to 50% for $n \geq 10$ in the heavy load case while the range is between 20% and 60% for the light load case. Figure 11 illustrates the approximation for the light load case (on the left) and the bound for the heavy load case (on the right) when $n = 10$. In each case we also plot the power savings computed via simulations. $\lambda = 0.05$ corresponds to a lightly loaded network while $\lambda = 4.0$ corresponds to a heavily loaded network. As we can see, the approximation for the light load case and the bound for the heavy load case match the simulation values well. The discrepancy we see is caused because of two reasons. In the heavy load case, our estimation of the number of RTS/CTS message exchanges is too optimistic. In the light load case, on the other hand, the discrepancy comes about for a different reason. To illustrate this reason, let us focus on sparse networks ($p = 0.1$). The expected degree of nodes in these networks is approximately 2 (for $n = 10$). The power savings computed by equation 1 is 20% while the savings produced by the simulation is 22%. This higher number comes about because of the specific random graph topologies that are generated in the simulation. For instance, consider the star network in Figure 12(a). The power savings obtained if the middle node transmits a packet is about 65% whereas the savings when any of the nine boundary nodes transmit is 0%. In Figure 12(c), on the other hand, the power savings are always exactly 20%. In Figure 12(b), the savings range between 0% (the leaf nodes) and 27% (the other nodes). Thus, the savings depend heavily on the degree of the transmitting nodes but the approximation works with the average degree only!

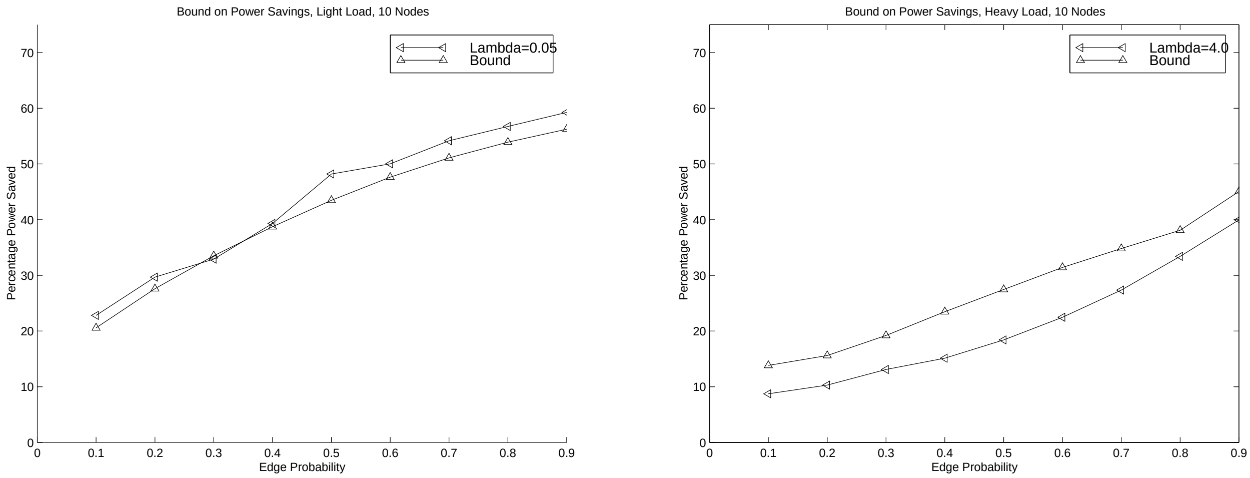

Figure 11: Bounds on Power Savings.

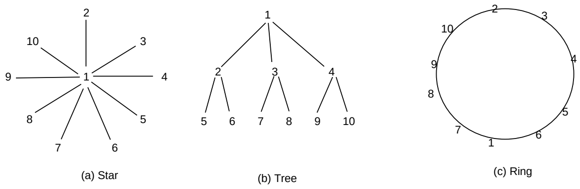

Figure 12: Three sparse networks.

4.4 Bounds for Power Savings for *all* random network topologies

In this section we will determine upper bounds for the *power savings for mobile ad hoc networks*. Here, as nodes roam about, the network topology changes constantly. Thus, we need to determine power savings *independent of the edge probability p*. As before, we consider the lightly loaded case separately from the heavily loaded case.

Theorem 4.5 *Under light load conditions, as $n \to \infty$, the power savings due to the PAMAS protocol is bounded by 75%.*

Proof: In the light load case, it is assumed that nodes that are transmitting are not nearby (unlike in the heavy load case). Consider a n node network with all possible topologies. Since there can be at most $n(n-1)/2$ edges, the number of distinct network topologies is $2^{n(n-1)/2}$. Our approach to finding the power savings is by considering a specific node, say t, as the transmitter with exactly k neighbors, with one of these neighbors being the intended receiver. We can find the number of distinct n node network topologies with this specific transmitter t by selecting some k nodes among the $n-1$ nodes and requiring that an edge exists between node t and these k nodes. We require that there be no edges between t and the remaining $n-k-1$ nodes so that t has exactly k neighbors. So far, we have accounted for k edges to be present and $(n-k-1)$ edges to be absent in determining our topologies. Therefore, the number of distinct graphs with node t having exactly k neighbors is $2^{\frac{n(n-1)}{2}-(n-1)}$. The fraction of all possible graphs with this scenario is then simply $2^{-(n-1)}$. There are many ways to choose these k neighbors. Under light load conditions, $k-1$ neighbors can be powered off with one neighbor being the intended receiver. Amount of power saved is given by the fraction $(k-1)/(k+2)$ without control overhead[5]. We estimate the power consumed due to the control overhead as follows. Node t will send an RTS packet, k neighboring nodes receive this, one neighbor sends CTS which will be received by at least one node, and finally a busy tone sent by the receiver. With control overhead, the power savings due to PAMAS in a topology with a transmitter having k neighbors will be bounded[6]

[5]Power consumed when no node is powered off is $1 + 0.5 \times k = 0.5 \times (k+2)$. Power consumed when $k-1$ nodes are turned off is $1 + 0.5 = 0.5 \times 3$. The power savings are thus $0.5 \times (k+2-3) = 0.5 \times (k-1)$. The fraction of power saved is thus $(k-1)/(k+2)$.

[6]Since we do not know the degree of the receiver, the power consumed during the RTS/CTS phase is computed as follows: transmit energy consumed $= (1+1+2) * 0.0625$ (1 RTS, 1 CTS and a Busy Tone that is twice as long), the receive energy consumed $\geq$

by $(k-1)/(k+2+(11+k)*.0625)$, where the factor .0625 is the amount of energy consumed in transmitting 32 bytes. We can calculate a bound on power savings for all possible random topologies under light load conditions to be,

$$\sum_{k=1}^{n-1} \binom{n-1}{k} \times 2^{-(n-1)} \times \frac{(k-1)}{(k+2+(11+k)*.0625)}$$

As expected, the power savings over all topologies increases with n and, in the limit as $n \to \infty$, the power savings is bounded by 75%. Clearly, transmitters with high degree result in greater power savings during data transmission but, simultaneously, consume more power during the RTS/CTS phase. □

Let us now consider the heavy traffic load situation where as many nodes as possible will be transmitting simultaneously without interference.

Theorem 4.6 *Under heavy load conditions, the power savings due to the PAMAS protocol is bounded by about 30% to 40% for n ranging from 10 to 20.*

Proof: Under heavy load, we expect transmitting nodes to compete for channel access. For each transmitting node there will be an intended receiver. Therefore, the number of simultaneous transmitters can vary from 1 to $n/2$ as we need an equal number of receivers. Consider the situation that in a random network, there are k simultaneous transmitters. For each transmitter, an edge must exist from the transmitting node to its intended receiving node and there cannot be any edges from that receiving node to the other $k-1$ transmitting nodes to avoid interference. Thus, with k transmitters, we must have one edge from each of these k transmitters to some k receiving nodes and the remaining possible $k(k-1)$ edges from the transmitter set to the receiver set must be absent, as otherwise there will be interference. So far, we have accounted for $2k$ nodes as either transmitting nodes or receiving nodes. To make the remaining $(n-2k)$ nodes not be able to transmit, we require that each of these nodes have an edge to at least one of the k receivers. There are $k(n-2k)$ possible ways of choosing these additional $(n-2k)$ edges to ensure we have maximized the set of transmitters. The transmitter set can be paired with the receiver set in $k!$ ways.

We can find the number of distinct graphs with exactly k transmitters as,

$$\beta = \sum_{k=1}^{k=n/2} \binom{n}{k} \times \binom{(n-k)}{k} \times k! \times k(n-2k) \times 2^{(n(n-1)/2-k^2-(n-2k))}$$

The kth term in the above summation has graphs with a maximum of k transmitters. We calculate the above sum with the kth term multiplied by k to find the total number of transmitters possible in these graphs. That is,

$$\gamma = \sum_{k=1}^{k=n/2} k \times \binom{n}{k} \times \binom{(n-k)}{k} \times k! \times k(n-2k) \times 2^{(n(n-1)-k^2-(n-2k))}$$

Now, if we divide γ by β we obtain the average value for the maximum number of transmitters that can simultaneously transmit in a random graph. We find this value to be

$$\Phi = \gamma/\beta$$

Therefore, we obtain the average power savings for heavy load case to be (see the derivation of eq. 3),

$$(n-2\Phi)/(n+\Phi) \tag{5}$$

We will include the control overhead for the channel by using the average degree information for a transmitters. With n nodes, the average degree of a node by considering all possible graphs is $(n-1)/2$. In the previous section, we derived an expression for energy e consumed in control overhead with a transmitter having d neighbors.

Therefore, we obtain the average power savings under heavy load to be

$$(n-2\Phi)/(n+\Phi+e\Phi) \tag{6}$$

We calculated this bound for different values of n and it varies from 30% to 40% for n ranging from 10 to 20. □

$0.5*(k+1+2)*0.0625$ since the RTS is received by all of t's neighbors, the CTS and Busy Tone are received by t (these are also received by the receiver's neighbors but we are ignoring this – hence we obtain an upper bound). Adding these quantities yields, total energy consumed for control messages $\geq 0.0625*(4+k/2+3/2)$.

5 Extensions to PAMAS

From the discussion thus far, it is clear that the PAMAS protocol has very good power conserving behavior. However, it is also clear that the ideas of power awareness that we have developed can be used to make other multiaccess protocols power conserving as well. This is because nodes are powered off only when they are blocked from transmitting or receiving. Thus, the delay characteristics of these protocols will not change. These ideas are discussed further in the next subsection where we outline how power awareness can be added onto protocols such as FAMA, MACAW, and others discussed earlier in section 2. Then, in the following subsection, we discuss possible extensions to PAMAS to improve its power conserving behavior and extensions to handle *broadcasts*.

5.1 Using Power Awareness in other Multiaccess Protocols

In order to conserve power, PAMAS powers off the radio interface in the event that a node is unable to either transmit or receive a packet. In section 3.1 we discussed how a node knows when and for how long to power off. If we are to incorporate power awareness in other multiaccess schemes [5, 12, 20] we need to develop similar protocols that will enable a node to determine when and for how long it needs to power off. Interestingly, however, it turns out that using the same channel for signalling and data limits the extent of power awareness that can be built into these protocols.

Based on our discussion in section 3.1, it is clear that, ideally, a node ought to power off either if it has no packets to transmit and a neighbor is transmitting or if at least one neighbor is transmitting and another is receiving. However, in order to implement this idea, we need to develop a *protocol* for powering off that answers the following questions based on feedback received from the radio channel:

1. When does a node power off?

2. For how long does it power off?

3. What happens when a node powers on and sees an ongoing data transmission?

Let us first look at the FAMA[12] protocol. Here, a node is in the *Passive* state if it does not hear anything on the channel and does not have a packet to transmit. If it receives a packet to send in the *Passive* state, it transmits a RTS and transitions to the *RTS* state awaiting a CTS. If no CTS arrives, it enters a *BACKOFF* state. It stays here for the appropriate interval, and if it does not hear anything on the channel for the entire period, upon coming out of backoff, it transmits a RTS. If it hears a transmission, it goes into the *Remote* state. A station transitions from the *Passive* state to the *Remote* state upon hearing a transmission or noise. In the *Remote* state, the node node waits for a time period before returning to the *Passive* state. The time period is determined as follows:

- If the station hears a RTS, it waits for the time needed to transmit a CTS plus the start of a packet. If it does not hear anything after this time, it goes back to the *Passive* state (or transmits a RTS if it has a packet to send).

- If it hears noise it waits for the time to send a maximum sized data packet. If it hears a CTS, it waits for the time required to send the data packet.

In the *Passive* state, there is no need to power off the node because it is not expending power receiving a transmission. However, power savings can be obtained by turning off the node when it is in the *Remote* state. In the first case, when the node hears an RTS and hears the start of packet transmission, the node could power off for the duration of the transmission (since it knows the packet length). Likewise, if it hears a CTS/noise, it can power off for the appropriate time period. It is clear that the delay and throughput behavior of FAMA does not change with these modifications while its power conserving properties do improve. However, the power savings are not the best possible. This is because, when a node powers back on, it may continue hearing noise/transmissions. This can happen if a node has two or more neighbors (who are not neighbors of each other) who begin transmissions at different times. When the node powers back on, it will not know the remaining length of the current transmissions (this is true even if it had not powered off because it will not hear the CTSs due to collisions with ongoing packet transmissions). Therefore, it will have to remain powered on until the transmissions complete. Observe that under heavy load conditions, this situation will occur frequently resulting in relatively poor power savings. In contrast, in our protocol, the node transmits *probe* packets (on the control channel) to determine the additional length of time for which it can power off.

In the MACA[17] and MACAW[5] protocols, a node enters the *Quiet* state when it hears a RTS or a CTS. In the former case, it waits for the packet transmission to begin and once started, it waits for the packet transmission to end. In the latter case, it waits for the packet transmission to end before transitioning out of this state. In either case, it makes sense to power off the node for the duration of the packet transmission. As in the FAMA, however, if new transmissions begin while a node is waiting for an ongoing one to complete, it does not know when the new transmission(s) will end. Thus, once it powers on after staying powered off for the duration of the first transmission, it will have to remain powered on until all the current transmissions finish. This results in needless power consumption. Finally, the MACA/PR [20] protocol can also be modified in a similar fashion (since it is based on MACAW and FAMA) but it suffers from the same drawbacks when it comes to conserving power.

The clustering mechanisms used in [14, 22], in contrast to the schemes discussed above, are far more amenable to power savings. Clustering divides the network into distinct components and a different spreading sequence is used for transmission within each cluster. Transmission within a cluster is accomplished using TDMA. Thus, every node is assigned a slot for transmitting its packets (modifications to the basic TDMA allow for the implementation of QoS guarantees). As such, the TDMA scheme is not amenable to power saving because a node cannot really power off (some node may be transmitting a packet to it). However, if we add minislots to the start of each TDMA cycle, we can implement power awareness as follows. For each node in the cluster, allocate a one-bit minislot. All the minislots are set to zero initially. If a node A wants to transmit to node B, A sets the bit in B's minislot. Thus B will remain powered on for the duration of the TDMA cycle. If, on the other hand, a node's minislot has not been set and it does not have a packet to transmit, it powers itself off for the length of the TDMA cycle.

5.2 Enchancements to PAMAS

Several enhancements are possible to the basic PAMAS protocol we have described. In this section we outline some of the more obvious ones. The first modification would be to add ACKs as has been done in MACAW[5]. Thus, the receiver transmits an ACK when a packet is received correctly. In addition, if the sender does not receive the ACK and transmits a RTS with the same packet number again, the receiver responds with an ACK instead of a CTS. Another modification that will improve throughput is to allow a node to transmit *multiple* packets when it has acquired the channel. This will serve to reduce time spent on channel access (but may increase delays). In order to implement power savings, we will need to ensure that the RTS/CTS/Busy Tone messages include the *total* length of the transmission (length of all packets being transmitted). Thus, a node will know the length of time for which it can power off.

Another possible enhancement to PAMAS is to power off the data interface of a node when its signalling interface is trying to acquire the channel. Thus, in a line subnetwork A-B-C-D, if C is transmitting to D while B is sending an RTS to A, powering off B's data interface ensures that C's transmission does not result in power consumption at B.

5.2.1 Support for Broadcasting

Broadcasting is often necessary in networks and, even though broadcasting is typically a network layer function, it is often necessary to provide MAC layer support. In PAMAS and other ad hoc network MAC layer protocols, a sequence of message exchanges (RTS-CTS) preceedes transmission of a packet. This message exchange ensures that the receiver is ready to receive the transmission. However, if a node needs to broadcast a message to all its neighbors, using the RTS-CTS sequence is meaningless because all the neighbors are potential receivers of the broadcast. If they all respond with a CTS (or if some respond with a CTS and others with a Busy Tone), the transmitter will hear noise and will be unable to decide what to do.

In PAMAS, the transmitter transmits a RTS_B message when it needs to transmit a broadcast. As in the basic PAMAS protocol, however, it transmits this message if no neighbor is transmitting or is scheduled to transmit. Upon hearing the RTS_B, a node receiving another transmission responds with a Busy Tone. Nodes that are free to receive the broadcast *do not respond*. If the transmitter does not hear any response in time equal to one roundtrip time plus processing delay, it transmits the broadcast message. If it hears a Busy Tone or noise in response to the RTS_B, it refrains from transmitting. It waits for the ongoing transmission to end (i.e., it waits for a maximum packet transmission time in case it heard noise or for the length of time specified in the Busy Tone) and retries.

When a node begins receiving the broadcast packet, it transmits a Busy Tone to warn other neighbors to refrain from transmitting.

A potential problem in this protocol is that a broadcast may collide with another transmission at some receiver. This is because nodes do not transmit CTSs in response to a RTS_B. Thus, their neighbors, two hops away from the transmitter of the broadcast, are unaware that it is about to receive a broadcast. If a node that heard a RTS_B heard noise when it expected the broadcast, it waits for the transmission to cease and transmits a NACK_B packet to the sender over the data channel (after the usual RTS-CTS exchange). *We leave recovery from this situation to the network layer because the network layer is aware of the network topology and is in the best position to decide whether the broadcast need be repeated.*

Power awareness can be easily incorporated here as follows. If we assume that every broadcast packet has a unique identifier (that is included in the RTS_B message) then a neighbor of the transmitter who has already received the packet can power itself off for the duration of the transmission.

6 Conclusions

In this paper we developed a novel multiaccess protocol for ad hoc networks that conserves power by turning off radios under certain conditions. We implemented and measured the performance of this protocol and showed that power savings range from 10% (in cases where the network is sparsely connected) to almost 70% in fully-connected networks. The noteworthy aspect of our protocol is that it achieves these power savings without affecting the delay or throughput behavior of the basic protocol. Finally, we discussed the applicability of our power saving ideas to other multiaccess protocols and showed how our ideas may be easily incorporated into these protocols, again without affecting their delay/throughput performance.

References

[1] Rooftop Communications,

`http://www.rooftop.com`

[2] `http://www.global-defence-review.com/DigitalBattlefield.html`

[3] `http://www.networks.digital.com/npb/html/products_guide/roamwir2.html`

[4] D. Beyer, "Accomplishments of the DARPA SURAN Program", *Proc. IEEE MILCOM'90*, Monterey, CA, Oct. 1990.

[5] V. Bharghavan, A. Demers, S. Shenkar and L. Zhang,"MACAW: A Media Access Protocol for Wireless LANs", *Proceedings ACM SIGCOMM'94*, pp. 212-225, 1994.

[6] Béla Bollobás, Random Graphs, Academic Press, 1985.

[7] A. Chandrakasan, T. Simon, J. Goodman and W. Rabiner,"Signal Processing for an ultra low power Wireless Video Camera", *3rd International Workshop on Mobile Multimedia Communications*, Princeton, NJ, September 25-27, 1996.

[8] A. Chockalingam and M. Zorzi, "Energy Consumption Performance of a class of Access Protocols for Mobile Data Networks", *Proc. IEEE VTC'98*, Ottawa, Canada, May 18-21, 1998.

[9] B. H. Davies and T. R. Davies, "The Application of Packet Switching Techniques to Combat Net radio", *Proceedings of the IEEE*, Vol. 75(1), January 1987, pp. 43-55.

[10] F. Douglis, F. Kaashoek, B. Marsh, R. Caceres, K. Lai and J. Tauber, "Storage Alternatives for Mobile Computers", *Proc. 1994 Symposium on Operating Systems Design and Implementation*, OSDI, November 1994.

[11] W. C. Fifer and F. J. Bruno, "The Low-Cost Packet Radio", *Proceedings of the IEEE*, Vol. 75(1), January 1987, pp. 33-42.

[12] Chane L. Fullmer and J.J. Garcia-Luna-Aceves,"Solutions to Hidden Terminal Problems in Wireless Networks", *Proceedings ACM SIGCOMM'97*, Cannes, France, Sept. 14-18, 1997.

[13] J.J. Garcia-Luna-Aceves, Chane L. Fullmer and Ewerton Madruga, "Wireless Mobile Internetworking", Manuscript.

[14] Mario Gerla and J.T.-C. Tsai,"Multicluster, Mobile, Multimedia Radio Network", *ACM-Baltzer Journal of Wireless Networks*, Vol. 1, No. 3, pp. 255-265, 1995.

[15] E.P. Harris and K.W. Warren,"Low Power Technologies: A System Perspective", *3rd International Workshop on Mobile Multimedia Communications*, Princeton, NJ, September 25-27, 1996.

[16] J. Jubin and J. D. Tornow, "The DARPA Packet Radio Network Protocols", *Proceedings of the IEEE*, Vol. 75(1), January 1987, pp. 21-32.

[17] P. Karn,"MACA – a New Channel Access Method for Packet Radio", in *ARRL/CRRL Amateur Radio 9th Computer Networking Conference*, pp. 134-140, 1990.

[18] B. M. Leiner, D. L. Neilson and F. A. Tobagi (Eds.), *Proceedings of the IEEE*, Vol. 75(1), Special Issue on Packet Radio Networks, January 1987.

[19] K. Li, R. Kumpf, P. Horton and T. Anderson,"A Quantitative Analysis of Disk Drive Power Management in Portable Computers", *Proceedings 1994 USENIX*, San Francisco, CA, pp. 279-291, 1994.

[20] Chunhung Richard Lin and Mario Gerla,"Asynchronous Multimedia Multihop Wireless Networks", *Proceedings IEEE INFOCOM'97*, (1997).

[21] M. J. Karol, Z.Liu and K. Y. Eng, "Distributed-queuing request update multiple access (DQRUMA) for wireless packet (ATM) networks, *Proc. IEEE ICC'95*, June 1995, pp. 1224-1231.

[22] C. R. Lin, M. Gerla, "Adaptive Clustering for Mobile Wireless Networks," IEEE Jour. Selected Areas in Communications, pp. 1265-1275, Sept. 1997.

[23] W. Mangione-Smith, P. S. Ghang, S. Nazareth, P. Lettieri, W. Boring and R. Jain, "A low power architecture for wireless multimedia systems: Lessons learned from building a power hog", *Proc. 1996 International Symp. on Low Power Electronics and Design*, Monterey, CA, pp. 23-28.

[24] S. Murthy, J. J. Garcia-Luna-Aceves, "A routing protocol for packet radio networks," Proc. MOBICOM'95, pp. 86-94. Nov. 1995.

[25] D. Petras and A. Krämling, "MAC protocol with polling and fast collision resolution for an ATM air interface", *Proc. IEEE ATM'96 workshop*, San Francisco, CA, Aug. 1996.

[26] Christian Röhl, H. Woesner and A. Wolisz, "A Short Look on Power Saving Mechanisms in the Wireless LAN Standard Draft IEEE 802.11", *Proc. of the 6th WINLAB Workshop on Third Generation Wireless Systems*, March 1997.

[27] J.E. Rustad, R. Skaug and A. Aasen, "New Radio Networks for Tactical Communication", *IEEE Journal on Selected areas in Communications*, Vol. 8(5), June 1990, pp. 713-727.

[28] A. Sen, M. L. Huson, "A New Model for Scheduling Packet Radio Networks," Wireless Networks, Vol. 3, No. 1, March 1997.

[29] Cheng-shong Wu and Victor O.K. Li,"Receiver-Initiated Busy-Tone Multiple Access in Packet Radio Networks", *Proceedings ACM SIGCOMM'87 Workshop*, Stowe, Vermont, Aug. 11-15, Vol. 17(5), pp. 336-342, 1987.

[30] Krishna M. Sivalingam, M. B. Srivastava and P. Agrawal,"Low Power Link and Access Protocols for Wireless Multimedia Networks", *Proc. IEEE Vehicular Technology Conference VTC'97*, Phoenix, AZ, May 4–7, 1997.

[31] Krishna M. Sivalingam, M. B. Srivastava, P. Agrawal and J-C. Chen, "Low-Power Access Protocols Based on Scheduling for Wireless and Mobile ATM Networks", *Manuscript*, http://www.eecs.wsu.edu/~krishna.

[32] W. Mangione-Smith and P.S. Ghang,"A low power medium access control protocol for portable multi-media systems", *3rd International Workshop on Mobile Multimedia Communications*,September 25-27, 1996.

[33] M. Stemm and P. Gauthier and D. Harada,"Reducing power consumption of network interfaces in hand-held devices",*3rd International Workshop on Mobile Multimedia Communications*,September 25-27, 1996.

[34] F. A. Tobagi and L. Kleinrock, "Packet Switching in radio channels: Part II - the hidden terminal problem in carrier sense multiple-access modes and the busy-tone solution", *IEEE Trans. Communications*, Vol. COM-23(12), 1975, pp. 1417-1433.

[35] H. S. Wilf, Algorithms and Complexity, Prentice Hall, 1986.

[36] S. Zdonik, M. Franklin, R. Alonso and S. Acharya,"Are "disks in the air" just pie in the sky?", *IEEE Workshop on Mobile Computing Systems and Applications*, Santa Cruz, CA, pp. 12-19, December 1994.

[37] Michele Zorzi and R. R. Rao, "energy Management in wireless Communications", *Proc. 6th WINLAB Workshop on Third Generation Wireless Information Networks*, March 1997.

Appendix A

The value of the average degree d of a random graph with edge probability p can be derived as follows. Let P_n denote probability that a n node network (with edge probability p) is connected. Let $P_{n/k}$ denote the probability that a node in the n node connected network has degree k. That is,

$$\begin{aligned} P_{k/n} &= \text{Prob[A node has degree } k \text{|the } n \text{ node network is connected]} \\ &= \frac{\text{Prob[degree of node is } k \text{ AND } n\text{-node network is connected]}}{P_n} = \frac{P_{n,k}}{P_n} \end{aligned} \tag{7}$$

We can then write d as,

$$d = \sum_{k=1}^{k=n-1} k P_{k/n}$$

In order to obtain the value of $P_{k/n}$ we compute P_n and $P_{n,k}$ as follows. To determine P_n, we construct a n node random network by adding a node to a $n-1$ node random network. Edges are put in as dictated by edge probability p. Using this model, we can write P_n as,

$$P_n = \sum_{i=1}^{n-1} P^i \tag{8}$$

where,

$$P^i \triangleq P[(n-1) \text{ node graph has } i \text{ connected components AND } n\text{th node connects them all}]$$

We can drop all terms above except the first two (i.e., P^1 and P^2) and still obtain a reasonable approximation for P_n. We thus obtain,

$$\begin{aligned} P_n &\approx P^1 + P^2 \text{ where,} \qquad (9) \\ P^1 &= (1-(1-p)^{n-1})P_{n-1} \\ P^2 &= \sum_{n_1=1}^{\lfloor \frac{n-1}{2} \rfloor} \binom{n-1}{n_1} (1-(1-p)^{n_1})(1-(1-p)^{n-n_1-1}) \times P_{n_1} P_{n-n_1-1} \end{aligned}$$

To derive $P_{n,k}$ we again assume that the $(n-1)$ node network has $i \leq k$ connected components and that the nth node connects them all. In addition, the nth node has a degree of k. We thus write,

$$P_{n,k} = \sum_{i=1}^{k} P_k^i \tag{10}$$

where,

$$P_k^i \triangleq P[(n-1) \text{ node graph has } i \text{ components AND } n\text{th node connects them AND n}th \text{ node has degree } k]$$

We again drop all but the first two terms and obtain,

$$\begin{aligned} P_{n,k} &\approx P_k^1 + P_k^2 \text{ where,} \\ P_k^1 &= \binom{n-1}{k} p^k (1-p)^{n-k-1} P_{n-1} \\ P_k^2 &= \sum_{n_1=1}^{\lfloor \frac{n-1}{2} \rfloor} \binom{n-1}{n_1} Q(n, n_1, k) P_{n_1} P_{n-n_1-1} \end{aligned} \tag{11}$$

The term Q in the expression above denotes the probability that the nth node has degree k and that it has at least one edge to each of the two components.

$$Q(n, n_1, k) = \begin{cases} \binom{n-1}{k} p^k (1-p)^{n-k-1} & \text{if } n_1 < k, n - n_1 - 1 < k \\ \sum_{k_1=1}^{k-1} \binom{n_1}{k_1} \binom{n-n_1-1}{k-k_1} p^k (1-p)^{n-k-1} & \text{if } n_1 \geq k, n - n_1 - 1 \geq k \\ \sum_{k_1=1}^{n_1} \binom{n_1}{k_1} \binom{n-n_1-1}{k-k_1} p^k (1-p)^{n-k-1} & \text{if } n_1 < k, n - n_1 - 1 \geq k \end{cases}$$

We can use the above two equations eq. 9 and eq. 11 to determine $P_{k/n}$ and hence d.