"When God closes all the doors, somewhere he leaves a little window open."

University of Alberta

A CONTROLLED FLOODING APPROACH TO EFFICIENT ROUTING IN AD-HOC WIRELESS NETWORKS

by

A.K.M. Ashikur Rahman 

A thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment of the requirements for the degree of **Doctor of Philosophy**.

Department of Computing Science

Edmonton, Alberta
Fall 2006

Your file Votre référence
ISBN: 978-0-494-23099-2
Our file Notre référence
ISBN: 978-0-494-23099-2

Abstract

The focus of this dissertation is ad hoc wireless networks. A correct efficient operation of such networks depends on the interaction of several protocols dealing with routing, medium access control, power control and many other issues. This dissertation is primarily concerned with the development and evaluation of such protocols.

In this dissertation, we show how flooding can be adopted as a reliable and efficient routing scheme in ad-hoc wireless mobile networks. It turns out that, with the assistance of some tunable heuristics, flooding is not necessarily inferior to sophisticated point-to-point forwarding schemes. We have developed a reactive broadcast-based ad-hoc routing protocol in which flooding exhibits a tendency to converge on a narrow strip of nodes along the shortest path between source and destination. The width of this strip can be adjusted automatically or by the user. We also point out a certain deficiency inherent in the IEEE 802.11 family of collision avoidance schemes in handling broadcast packets, and show how to fix it to provide better service to broadcast-based routing schemes represented by our variant of controlled flooding.

Next, we consider the topology control problem whose objective is to minimize the amount of power needed to maintain connectivity. The issue boils down to selecting the optimum transmission power level at each node, based on the position information of reachable nodes. Local decisions regarding the transmission power level induce a subgraph of the maximum powered graph. We propose a new algorithm for constructing minimum-energy path-preserving subgraphs of the maximum powered graph.

Routing protocols use another important service called 'broadcasting' for different purposes. The primary goal of any broadcast scheme is to reduce the total number of retransmissions needed to reach all nodes in the network. Another performance measure, which has not received as much attention, is the broadcast latency. We demonstrate that these two objectives, i.e., reducing the number of retransmissions, and reducing the latency, are contradictory and result in a trade off. We also show how to adjust the stochastic component of the popular class of contention resolution schemes based on IEEE 802.11 to significantly reduce the broadcast latency.

Acknowledgements

I am indebted to many people for the completion of this dissertation. I earnestly thank my supervisor Dr. Pawel Gburzynski for his excellent guidance and advice. The immense trust he placed in my abilities was always a great source of motivation. I knocked his office door several times for his valuable advice without any prior notice, and he always welcomed me with a smile, even under tremendous work pressure.

I would also like to thank the members of my supervisory committee, specially, Dr. M.H. MacGregor and Dr. Janelle Harms, for providing me useful feedback and comments about the research. I would also like to thank the members of Communication Network Research group for providing me useful feedback and comments over the course of my research.

Finally, I am grateful to my parents, brother, sister, my loving wife and my eleven months old son for their valuable support and encouragement during the entire period of this research.

...to my Parents

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

The popularity of wireless technology is ever increasing. During the past several decades, many researchers from both academia and industry have made it a main focus of their research. The ever-increasing popularity of wireless networks suggests that next generation networking will be mainly based on wireless technology.

Two possible configurations of wireless networks are *infrastructured* and *infrastructureless*. Traditional cellular networks are an example of infrastructured networks. In a cellular network, mobile hosts communicate with a *base station* within their communication range. Base stations act as a bridge between two mobile hosts or between a mobile host and a fixed host. These cellular networks are limited by their need for a fixed infrastructure. Mobile ad-hoc networks are infrastructureless networks, where this limitation is completely eliminated.

Ad-hoc networks are collections of (possibly) mobile nodes that do not require any centralized control. Nodes cooperate with each other in forwarding packets over the network. Each node in the ad-hoc network is treated equally and can behave as a producer of data packets, consumer of data packets, or simply a passive forwarding intermediate node (i.e. a router).

Some characteristics and problems of ad-hoc networks are shared by all wireless networks, ad-hoc and infrastructure-based alike. For example, the following features are of general importance:

(1) *Dynamic Topology:* the nodes forming ad-hoc networks can move around in a generally unknown fashion, which tends to produce essentially unpredictable topology changes.

(2) *Link quality and bandwidth constraints:* wireless links are subject to higher loss rates than wired ones, which can cause inherently higher delays or poorer quality of service than those occurring in wired networks. The available bandwidth is also much scarcer than in wired networks.

(3) *Limited energy:* the nodes are assumed to operate on batteries or other exhaustible energy sources that deplete over time.

(4) *Cooperative environment:* due to the lack of fixed infrastructure, ad-hoc networks are multi-hop in nature. Each node should actively participate in packet routing.

(5) *Limited security:* ad-hoc networks are distributed systems. As a result, they are subject to major security threats, such as eavesdropping, denial of service, data alteration, etc.

The specific characteristics of ad-hoc networks bring about new research problems, such as self-routing, addressing, channel access mechanism, topology control, power optimization, etc. Using the traditional layered design paradigm, a general trend to attack these problems is to first identify which problems are particularly suitable to be solved at a particular layer in the protocol stack and solve those problems within the boundary of that particular layer. As a result, a well-designed architecture for mobile wireless ad-hoc networks needs attention in every layer. Although there may exist a clear idea about which problem should be solved at which particular layer, this is not good enough. A particular problem may be solved more efficiently if we allow information sharing between layers. For example, information about a neighbor's position, link failure, etc. may be shared between the Medium Access Control (MAC) layer and the network layer (the routing layer). This will give both layers a better view of the network topology, which in turn will translate into a more efficient operation, e.g., the reduced overall number of messages in the network. In summary, strict protocol layering in wireless ad-hoc networks tends to be detrimental to performance. It is generally beneficial to all the aspects of network performance to address the critical goals of the network, such as routing, reliable medium access control, and power optimization globally, i.e., through the collaboration of different layers.

1.1 Motivation

Ad-hoc wireless networks will play an increasingly important role in both military and civilian environments where wireless access to a wired backbone is either ineffective, impossible or costly and difficult to build. For this reason mobile ad-hoc networks have been the focus of many recent research and development efforts.

The original motivations for ad-hoc network technology are found in military applications. In battlefields, where the territory is unknown, an infrastructured network is impossible to build and maintain. Ad-hoc networks can provide the required mobile communication platforms in a battle-field. Combinations of short- and wide-range ad-hoc networks can provide global, reliable coverage, even in highly adverse operating conditions.

Natural disasters (such as floods or earthquakes) may lead to crisis situations by destroying the entire communication infrastructure. Restoring communication in such circumstances is essential and should be done as quickly as possible. By creating ad-hoc networks, a quick, temporary infrastructure could be set up in hours, instead of the days of work required to rebuild the wired communication infrastructure.

The recent invention of small wireless devices such as personal digital assistants has opened up the opportunity for a number of commercial applications of ad-hoc networks. At a conference, users can spread and share information between each other by forming ad-hoc networks between notebook or palmtop computers. Short-range ad-hoc networks can be used for interaction among various mobile devices (e.g., a cell phone and a PDA) by replacing the need for irksome cables. Another commercial application might involve providing local coverage quickly at a remote construction site (for example in some parts of Canada's Northern Territories).

A sensor network is a subclass of wireless ad-hoc networks which is currently the subject of intensive research. Sensors can be easily deployed in remote places where they can monitor and/or control the physical environment. Coordinating their activities by providing them with a network infrastructure will revolutionize information gathering and processing in many situations.

All of the application scenarios mentioned above require special attention to some unique characteristics of ad-hoc wireless networks. In ad hoc wireless networks, nodes have limited power and memory, share a common wireless channel, move randomly in any direction and depend on each other for forwarding information. Since network nodes are mobile, the ad-hoc network must cope with dynamic topology. Network functions such as routing, medium access control, etc., must address the issue of dynamic volatile network topology. Ad-hoc nodes are supposed to be cheap and simple. The hardware they are running on should be readily available and inexpensive, and its power requirements should be minimal.

Our work at the routing layer is motivated by the difficulty in designing simple, low-cost, small-footprint, on demand routing protocols for ad hoc wireless networks. Many existing protocols presented in the literature are either complex in operation or costly in nature. For example, all position-based localized routing requires that every node know 1) the position information of all nodes in its neighbors, and 2) the position information of the final destination of the packet. The first requirement is fulfilled by the use of some special equipments such as Global Positioning System (GPS) devices, which obviously (and in many cases prohibitively) increase the hardware cost of the nodes. The second requirement is even more difficult to fulfill because it requires every node to propagate the changes in its position information over the entire network (this is known as *location service*). This location service is certainly a bandwidth intensive and poorly scalable operation.

Nodes in an ad-hoc wireless network are battery powered, tend to have low processing capabilities and limited memory. This is especially true in sensor networks: most of them make sense commercially only if the underlying hardware is extremely cheap. Despite the ever decreasing cost of microcontrollers, we see no reduction in the demand for the ones from the lowest end of the spectrum. On the contrary: their low cost and small power requirements enable new applications and trigger even more demand. As a result, it may be difficult for a node to maintain large routing tables needed by the well-known routing protocols, such as link-state or distance vector routing. For such nodes, a "nearly state-less" routing protocol would be more suitable. Routing protocols that do not

require the maintenance of routing tables representing network state are known as stateless routing protocols.

Flexibility is another important issue in the design of routing protocols. Some applications will run on hundreds of scattered stations, others just on a few. Intuitively, routing among the former group of applications will be more challenging. Due to the higher number of messages in the network, the protocol must be careful not to waste its scarce bandwidth resources. This, of course, comes with the price of increased memory requirements and computational complexity of the protocol. In the later group of applications running on a small number of nodes, virtually any reliable method of packet delivery from one station to another will do. Complexity of the protocol may be further reduced, which translates into lower memory requirements and lower cost. If the designer of an ad-hoc application could control, to some extent, the way the underlying protocol works, the performance of such a solution would be better. Allowing an application to dynamically influence the routing protocol may provide additional flexibility and improve the overall performance of an ad-hoc network. The applications we envision here will not be studied and developed in isolation from the networking paradigm, but are expected to interact with it, including the lower layers, for the benefit of their users. For example, the network layer may suffer poor performance due to a highly mobile environment. In this situation, applications developed on top of this layer need to be resilient and adaptive to maximize the quality of service under given circumstances.

Reliability is another important issue for efficient routing in ad-hoc wireless network. Since all the nodes use the same physical channel, medium access control (MAC) plays an important role in coordinating channel access among the nodes so that data packet gets through from one node to another. For such a MAC protocol, it is very important that it provides reliable data transmission service to the upper layers, otherwise, the end-to-end successful delivery of packets might require multiple unnecessary retransmissions causing unnecessarily longer delays and wasting valuable bandwidth resources. Over the years, many medium access control schemes have been proposed and, finally a standard named IEEE 802.11 has become popular and commercially available. Unfortunately the IEEE 802.11 family of collision avoidance schemes do not provide reliable service for broadcast packets. The virtual carrier sense mechanism is absent and, there is no provision for acknowledgments in the data link layer for broadcast packets. The poor handling of broadcast packets by the IEEE 802.11 family of collision avoidance schemes has motivated our work at the medium access control layer.

The performance of routing protocols also depends on the underlying topology of the ad hoc networks, because the underlying topology ultimately determines the choice of intermediate relay nodes to route the data packets. The network topology can be easily modified for optimal performance by choosing different transmission power levels at the different nodes in the network. The power optimization problem in ad-hoc wireless networks deals with techniques for selecting the appropriate transmission power at every node, such that the overall power consumption by the network

is minimized, and spread uniformly over all nodes. In addition to the obvious benefits, like extending the life time of a node, good strategies for selecting the transmission power have the potential to reduce interference/contention and thus increase the traffic carrying capacity of the network – by facilitating spatial reuse. This motivates our work on transmission power control.

The importance of transmission power control is most clearly visible in sensor networks. Once the sensors are placed in some, possibly remote, environment, they become detached from any possible maintenance service. It is thus expected that the power budget will be the major driving issue behind protocols designed for such networks, since a sensor's lifetime is defined by the lifetime of its battery. Transmission power control also affects other performance issues of the entire network. Transmission power level determines the range of transmission. As a result, the choice of power level affects the performance of the medium access control scheme because the extent contention depends on the number of nodes within the transmission range. Power control increases/decreases the number of hops needed to reach a destination which will have an ultimate effect on end-to-end delay. The choices of power level also determine the connectivity of the whole network and can significantly affect its traffic-carrying capacity.

Network-wide broadcasting is another important, fundamental and much needed service in many applications of ad hoc wireless networks. The popular reactive routing protocols such as DSR [22], AODV [43], TORA [41], depend on broadcasting for route discovery. Proactive protocols, such as DSDV [44] and WRP [36], periodically broadcast updated information about cost metrics. Some protocols even use (selective) broadcasting for the actual forwarding [48]. While designing a broadcast protocol for ad hoc wireless networks, one of the primary goal is to reduce the extent of collisions and retransmissions, specially redundant retransmissions, while reaching all the nodes in the network. Another important but (so far) unattended goal is to reduce the amount of time needed to complete a broadcast operation. Reducing the broadcast latency is very important because this will ultimately determine how quickly route discovery can be made. The importance of broadcasting protocols and its impact on performance of other protocols have motivated our work on the issue of low-latency broadcasting.

1.2 Contributions

In this dissertation, we focus on four problems. To solve these problems we assume that information sharing between different protocol layer does not have to follow the strict standard model (e.g., the 7-layer OSI stack). These are the four major contributions of the dissertation:

(1) *Broadcast-based routing:* We begin our work with the *routing* issue in ad-hoc wireless networks. Routing is typically a network-layer problem. We classify routing protocols into two categories: *unicast-based* and *broadcast-based.*

The class of unicast-based protocols (i.e. proactive and reactive protocols) identify the routes

to the destination before forwarding packets. Proactive protocols identify the routes by maintaining more or less complete and up-to-date routing information (routing tables), such that it is immediately available when needed, while reactive protocols achieve this goal via a separate route discovery procedure carried out upon demand. Once the route to the destination is established, the question of maintaining the up-to-date status of those paths naturally arises. Route maintenance is done either in pro-active fashion (by periodic information exchange) or in reactive fashion (by generating special control packets). From a proactive protocol's point of view, this maintenance phase is very costly because neighbors need to exchange their view of the whole topology with each other. Route maintenance is done on-demand in reactive protocols, which results in a lower overhead. However, reactive protocols are more vulnerable to frequent topology changes. With frequent changes in topology, the probability of broken routes increases, causing more frequent route maintenance to take place. The topology of the network may change not only due to "uncontrollable" factors such as node mobility, interference, and bad weather conditions; but also due to some "controllable" parameters such as transmission power and antenna direction. In summary, the class of unicast-based protocols may generate lots of overhead messages due to the nature of their complex set of operations such as route discovery and route maintenance. The route discovery phase can be very expensive in terms of communication costs. Explicit route maintenance can be even more costly as it needs the explicit communication of substantial routing information and the usage of limited memory.

On the other hand, the class of broadcast based protocols is mostly based on flooding. The concept of flooding is very simple, the sender broadcasts its message to all of its neighbors. Its neighbors, in turn, rebroadcast the packet to their neighbors and so on until the packet reaches every node in the network. When the topology of the network is very unstable and dynamic such that reliable routes are difficult to find, flooding is the only option for transmission. The interesting property which distinguishes flooding from other protocols is its ability to route packets without any kind of knowledge about the topology of the network. That means that it can operate in a memoryless fashion, and no control packets ever need to be generated in the network. On the downside, flooding 1) is bandwidth intensive, 2) can generate many redundant and superfluous packets, and 3) can increase the probability of collision and congestion.

Clearly the class of unicast-based protocols and broadcast-based protocols stand at the two opposite end of the spectrum. Our target is to find something "in-between" and propose a protocol that will try to take advantages of the "good-features" of both paradigms. In particular, it will try to take advantages of flooding's simplicity and avoidance of control messages by combining route discovery and data forwarding, while being much more bandwidth efficient than flooding. On the other hand it will be reactive in nature like some other unicast-based protocols and will not work in a complete "memory-less" fashion. Instead it will try to col-

lect and store very little information about the topology from the data packets passing by. In addition, it will provide some 'tunable' parameters to the upper layers so that its behavior and local resource usage (memory, amount of processing) can be suitably controlled based on the nature of the environment.

(2) *MAC support to enhance reliability of broadcast-based routing:*

The goal of our work on the medium access control layer is to enhance the reliability of broadcast-based routing. As a collision avoidance scheme, the IEEE 802.11 MAC protocol works well for unicast packets but is not necessarily adequate for broadcast packets. The "virtual carrier sense" mechanism is practically useless for broadcast packets due to multiple recipients. Broadcast packets rely upon the "physical carrier sense" mechanism for collision avoidance. The situation aggravates in the absence of acknowledgment because the sender cannot take a retransmission decision without any feedback from recipients. As a result, broadcast-based routing algorithms tend to suffer in performance when built on top of IEEE 802.11 MAC. The second contribution of this research is to enhance the reliability of broadcast-based routing. With an innovative idea of "fuzzy acknowledgments," and some cross-layer interaction between MAC layer and network layer, we show how to enhance the reliability of broadcast-based routing.

(3) *Power control for energy-efficiency:*

Next, this dissertation addresses the issue of reducing the power consumption of wireless devices in ad-hoc networks by designing a transmission power control algorithm based on close collaboration between the MAC layer and the routing (network) layer. We ask whether it is possible to design a network being an interesting subgraph of the maximum powered graph. This means that nodes will intentionally reduce their transmission power in a manner that will explore some interesting features of the resulting graph, induced by the smaller neighborhood coverage. These features include: energy minimization, path preservation, connectivity, low node degree, and invariant performance with respect to network dynamics.

The major contribution of our approach is the design of a new, easy to implement MAC protocol characterized by a small number of overhead messages needed to construct efficient minimum-energy paths. The collaboration between the routing layer and the MAC layer assumed in our algorithm consists in combining the objectives: the routing layer is responsible for discovering the optimal topology and the MAC layer is involved in optimizing the number of message exchanges to discover this topology. More specifically, the network topology established by our algorithm: 1) preserves connectivity, 2) preserves all the minimum-energy paths between every pair of nodes, 3) is able to accommodate mobility with an almost constant amount of overhead, 4) keeps the number of overhead messages needed to construct the topology under varying node density independent of the choice of suitable values of any

parameters needed by the algorithm.

(4) *Broadcast speedup:*

Finally, we study the issue of broadcasting from the viewpoint of the total amount of time needed to reach all nodes in the network. Not surprisingly, the problem of efficient broadcasting in the wireless ad-hoc environment has received considerable attention. Many efficient algorithms have been proposed, whose main goal is common: minimizing the number of retransmissions while ensuring that the broadcast message reaches all nodes in the network. However, little efforts were made to explicitly address the issue of reducing broadcast latency defined as the time elapsing from the moment a node initiates a broadcast until the last node in the network has received a copy of the message. This performance measure is hardly irrelevant. For example, in the case of a reactive routing protocol, the broadcast latency effectively determines how quickly routes are discovered, which is the primary quality criterion of the underlying routing scheme. This is especially true under conditions of non-trivial mobility, which triggers frequent route changes and, consequently, frequent invocations of the route discovery mechanism. For a proactive protocol, which disseminates the cost metric information via broadcasting, the high cost of this operation will tend to reduce the optimum update frequency and render small updates too costly to disseminate. Consequently, the network will tend to operate with inaccurate routing information, at a performance penalty that, under non-trivial mobility scenarios, may turn out to be unacceptable.

At first sight, one might naively assume that the objective of minimizing the number of retransmissions in a broadcast scheme automatically implies minimizing the latency: after all, we minimize the total time spent by all nodes on transmitting packets, so what else is there to gain? As we demonstrate in our study, this is not true. To conclude that a message should not be retransmitted, a node must (implicitly or explicitly) acquire some information from other nodes. This involves time and, in fact, puts the two objectives at the opposite ends of a trade-off.

The primary contribution of our work in this area is a proposed modification to the backoff component of the IEEE 802.11 family of collision avoidance schemes that effectively reduces the broadcast latency while keeping the number of retransmissions small.

1.3 Document Structure

The remaining part of this thesis is organized as follows. Chapter 2 provides an overview of those major accomplishments in the area of ad-hoc wireless networking that are related to our research. Chapter 3 describes our broadcast-based routing protocol called TARP (Tiny Ad-hoc Routing Protocol). In that chapter we also illustrate the deficiency of medium access control schemes based on IEEE 802.11 in supporting broadcast-based routing. It also discusses our proposed modification to

IEEE 802.11 aimed at improving the reliability of routing based on broadcast. Detailed simulation results are presented at the end of that chapter. Chapter 4 introduces our topology control algorithm and discusses its performance. Chapter 5 describes our work on reducing broadcast latency along with simulation results demonstrating its effectiveness. Finally Chapter 6 concludes this dissertation by pointing out some open research problems related to our work.

Chapter 2

Overview of past work

We are particularly interested in four problems:

(1) *Routing,*

(2) *Medium Access Control Layer support to enhance reliability of broadcast-based routing,*

(3) *Power control for energy-efficiency,*

(4) *Broadcasting.*

This chapter will provide a brief description of each of the problems and some useful research work already done on these four classical issues.

2.1 Routing

The role of routing in an ad-hoc wireless network is to provide peer-to-peer connectivity between any two nodes, possibly involving multiple hops through a sequence of intermediate nodes. Here are some important postulated features of such algorithms:

Distributed organization A centralized approach is not feasible for an ad-hoc networks due to its very nature: there is no centralized authority and the population of nodes is generally unknown.

Fast responsiveness (convergence) after link changes As ad-hoc networks are subject to unpredictable movement of nodes, link changes occur frequently. It is desirable that the routing protocol reacts to link changes as quickly as possible.

Localized reaction to topology changes The effect of topology changes should be completely localized and should not affect distant regions of the network.

Loop avoidance Discovered paths should avoid the formation of loops to enforce efficient routing.

Scalability The per node storage and communication cost should grow as a small function of the total number of nodes.

Routing protocols for ad-hoc wireless networks can be broadly divided into two groups—*proactive* and *reactive*. Proactive protocols try to maintain up-to-date routing information at every node in anticipation of demand. This is normally achieved by the periodic exchange of information between nodes. Reactive protocols collect the necessary routing information only when it is explicitly needed to sustain an actual session. A hybrid approach combining the best features of these two approaches, is also possible.

2.1.1 Proactive protocols

This section describes some popular existing proactive routing protocols. They differ with respect to the number of necessary routing tables and the methods by which changes in topology are propagated.

2.1.1.1 Destination-Sequenced Distance-Vector Routing (DSDV)

In DSDV [44], every mobile station maintains a routing table that lists all available stations, the number of hops to reach the station and the neighboring node to use to reach that station. Each entry is marked by a sequence number assigned by the destination to ensure that the routes are loop free. Updates to the routing table are both time-driven and event-driven. The stations periodically transmit their routing tables to their immediate neighbors at regular intervals (time-driven). A station also transmits its routing table if a significant change occurs in its table (event-driven). Upon receiving routing tables from its neighbors, a node uses the information with the most recent sequence number for updating purposes. If two updates arrive with the same sequence number then the route with the better metric is used. Nodes also keep track of *average settling time*, the weighted average time for which the route may fluctuate before receiving the route with the best metric. A node delays its broadcast of routing update until the settling time to reduce those unnecessary updates that would occur if a better route could be discovered in the near future.

Two major drawbacks of DSDV are that it exchanges a large number of packets between nodes to keep the routing tables up to date, and it reacts slowly to topology changes.

2.1.1.2 The Wireless Routing Protocol (WRP)

The Wireless Routing Protocol (WRP) [36], is a table-driven, distance-vector protocol. A node in the network maintains four tables: a *Distance table*, a *Routing table*, a *Link-cost table* and a *Message Retransmission List (MRL)*. The Distance table of a node s contains the distance of each destination d from each neighbor n of node s and the predecessor toward d reported by node n. The Routing table contains the shortest distance to each destination and the predecessor toward the destination and successor of the node in this path. It also contains a tag to indicate if the entry is a simple

path, a loop, or invalid. The Link-cost table contains the cost of each link to n's neighbors. The Message Retransmission List (MRL) keeps track of which updates need to be retransmitted and which neighbors should acknowledge the retransmission. Routing tables are exchanged between neighbors in the form of update messages either periodically or upon link changes. The update message contains a list of updates comprising the destination, the distance to the destination and the predecessor of the destination. It also contains a list of responses indicating which nodes should send an ACK. If a link is lost, the node detecting the lost link sends update messages to its neighbors. The neighbors then modify the distance table entry and check for new possible paths via other nodes. Any new paths are relayed back to the original node for an update. A unique feature of this algorithm is that it checks the consistency of the predecessor information reported by all its neighbors every time it detects link changes of any of its neighbors. This ultimately helps to avoid "count-to-infinity" problem and provides faster convergence.

2.1.1.3 Cluster-head Gateway Switch Routing Protocol (CGSR)

The CGSR [8] protocol differs from other proactive protocols in the way it organizes the network and issues addressing. Nodes are aggregated into clusters and one node per cluster controls the whole cluster. A distributed algorithm is used to elect the cluster head. All nodes within the communication range of a cluster head belong to that head's cluster. Two clusters are joined via a "gateway" node which is within the communication range of both cluster heads. Frequent node movement may require frequent runs of the cluster head election algorithm. To reduce cluster head selections, CGSR uses the Least Cluster Change (LCC) algorithm. According to the LCC algorithm, cluster head changes occur in only two cases: 1) if two cluster heads come within the range of each other 2) when a node moves out of the range of all cluster heads.

Routing in CGSR is based on DSDV and switches alternatively between cluster heads and gateways. The source of a packet sends the packet to its cluster head first. Then the cluster head sends it via a gateway node to another cluster head. This switching between cluster heads and gateways continues until the packet reaches the destination's cluster head and it is transmitted to the destination. Figure 2.1 shows an example of CGSR routing. To use this method effectively, each node must keep a "cluster member table" where the cluster head of every node in the network is stored. This table is then periodically broadcast using DSDV.

The main drawback of this protocol is the creation and maintenance of clusters. Also cluster heads and gateways are special nodes which may be overloaded and have higher power demands. Moreover, as it uses DSDV as the underlying scheme, its overhead is similar to that of DSDV.

2.1.2 Reactive protocols

Reactive protocols collect the necessary routing information only when it is explicitly needed to sustain an actual session. All reactive protocols maintain, to some degree, three separate phases:

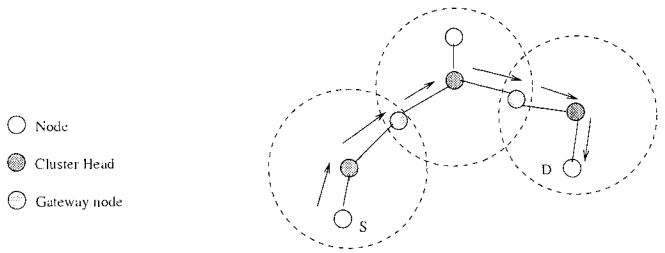


Figure 2.1: Routing from S to D in CGSR.

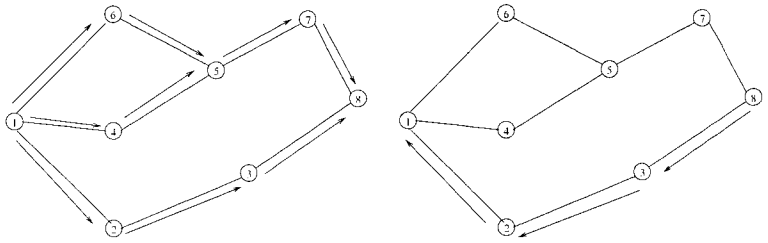
1) Route discovery, 2) Actual Data Transmission and 3) Route maintenance. The generic reactive approach is as follows: If a node needs to transmit data packets to certain destination, it will establish the route through a route discovery procedure. Then the data transmission begins over the established path. As the session goes on, the established path may cease to exist due to mobility in the network, or node failure. As a result, reactive protocols also need route maintenance procedures incorporated into them.

2.1.2.1 Ad-hoc On-Demand Distance Vector Routing (AODV)

In AODV [43], a source initiates a path discovery process by broadcasting a RREQ packet. All the receiving nodes will rebroadcast the RREQ packet in turn if they do not have any fresh routing information about the destination. This process continues until the RREQ packet reaches the destination or an intermediate node which has fresh route information. When a node receives a RREQ packet, it keeps a record of the address of the neighbor from which it received the first copy. This establishes a reverse path. All subsequent copies are discarded. Every RREQ has a unique id. With the combination of initiator's IP address and the unique id, all RREQ packets can be distinguished from each other. This eases the process of avoiding duplicates. When an RREQ reaches its destination or an intermediate node having valid route information, it sends back an RREP message. The RREP message follows the path that was used by the RREQ packet in the reverse direction. A node receiving the RREP again keeps record of the neighbor from which it received the RREP, establishing the forward path. AODV uses the same concept of the destination sequence number as DSDV to avoid loop-formation. Figure 2.2 shows the route discovery procedure in AODV.

In addition to route discovery, AODV has a route maintenance procedure. If a source moves, it will re-initiate the route discovery procedure if the route to a destination is still needed. If a node along the established path moves, then its neighbor along the path will detect a link failure. Upon detecting a link failure, a node will inform all of its upstream neighbors which are currently using this node actively to transmit to a certain destination. This process will continue until all the upstream nodes have been notified (including the original source). All upstream nodes can remove this path information, and the source may re-initiate another route discovery.

AODV requires symmetric links between nodes and cannot utilize routes with asymmetric links.



(a) (b)

Figure 2.2: Route Discovery process in AODV (a) Propagation of RREQ, (b) Path taken by RREP.

AODV does not keep track of multipaths. When a route fails a source needs to restart the route discovery process.

2.1.2.2 Dynamic Source Routing (DSR)

DSR [22] uses a source routing technique whereby the original sender of the data packet explicitly lists the complete sequence of nodes through which the packet is to be forwarded. Initially, when the source node wants to send a packet to a certain destination, it checks in its routing cache to see whether it already contains the route to the destination. If it does not find an entry it will initiate a route discovery process by broadcasting a *route request* packet. This route request packet contains the source address, destination address and a unique id. Each node receiving the route request will check if it has a valid route entry for the destination. If it does not have any valid entry, it will append its address in the *route record* of the route request and will rebroadcast the request. A node will ignore any duplicate route request as well as those with its own id already in the route record (thereby avoiding the formation of loop). This process will continue until the route request reaches either the destination or an intermediate node which has a valid route to the destination. In both cases a *route reply* packet is sent back to the originator of the route request. If the node sending the route reply is the destination itself then it will copy the route record from the route request to the route reply. If it is an intermediate node then it will append its cached route to the destination to the route record and put it in the route reply. The route reply can be sent in three different ways. If the responding node has a cached route to the initiator, it can use it to send the reply. If symmetric links are assumed then the responding node can simply reverse the route record in the route request and use it to send route reply. If neither is possible then the node will initiate a separate route request process and piggyback the route reply into it. Figure 2.3 shows the route discovery procedure in DSR.

Many optimizations to DSR were proposed based on aggressive-caching and analyzing the cached information. For instance, each data packet contains the complete route which can be extracted by intermediate nodes so that they can learn the routes to all downstream nodes in the path. Additional routes can be constructed by combining two or more cached routes. More topology

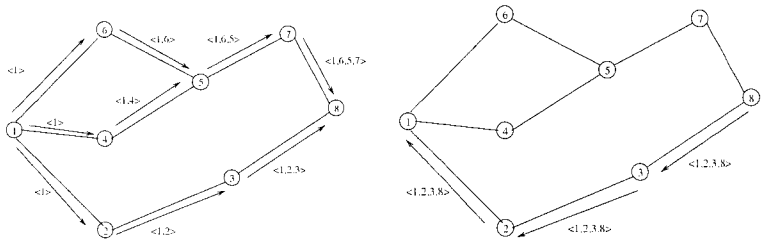


Figure 2.3: Route Discovery process in DSR (a) Propagation of route request, (b) Propagation of route reply.

information can be gathered by simply overhearing the routes used by the neighboring nodes.

Route maintenance also uses cached information. A broken link is detected when a node encounters a fatal transmission problem or does not receive an (possibly passive) acknowledgment. Upon a link failure, a *route error* packet is generated. When a node receives a route error message it removes the hop in error and all routes containing the hop are truncated accordingly.

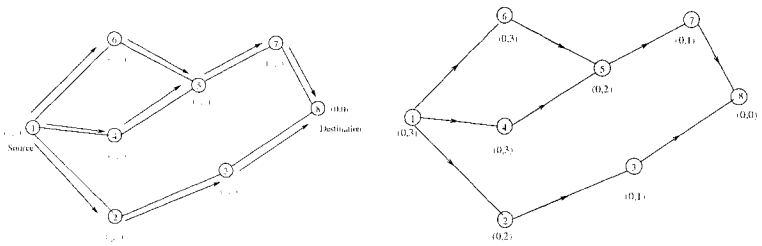
On the downside, DSR's technique of specifying the source route in every packet incurs a rather serious overhead. Also the use of aggressive caching may lead to stale route information being injected in the network. Overall, DSR involves a lot of processing due to its aggressive caching.

2.1.2.3 Temporarily Ordered Routing Algorithm (TORA)

TORA [41] is designed to localize the reaction to topology changes to the set of nodes affected by a change. This localization technique helps TORA to be very useful in highly dynamic environments. Another advantage of TORA is that it maintains multiple routes to destinations.

The route creation in TORA is based on the concept of "heights" with respect to a certain "reference level." These height assignments are used to construct a Directed Acyclic Graph (DAG) rooted at a destination. Thereafter, each link is assigned a direction (upstream/downstream) based on the heights of the nodes it is connecting. The route construction procedure is roughly as follows. When a node needs a route to a destination, it broadcasts a *route request* message. The request is rebroadcast until it reaches the destination. The destination then assigns itself a zero height. Then the destination broadcasts an update message including its height of zero. Each node receiving the update message will assign its own height as one greater than the height mentioned in the update message. The node then rebroadcasts the update message replacing the received height with its own height. This creates a DAG rooted at the destination. Figure 2.4 shows the DAG construction process in TORA.

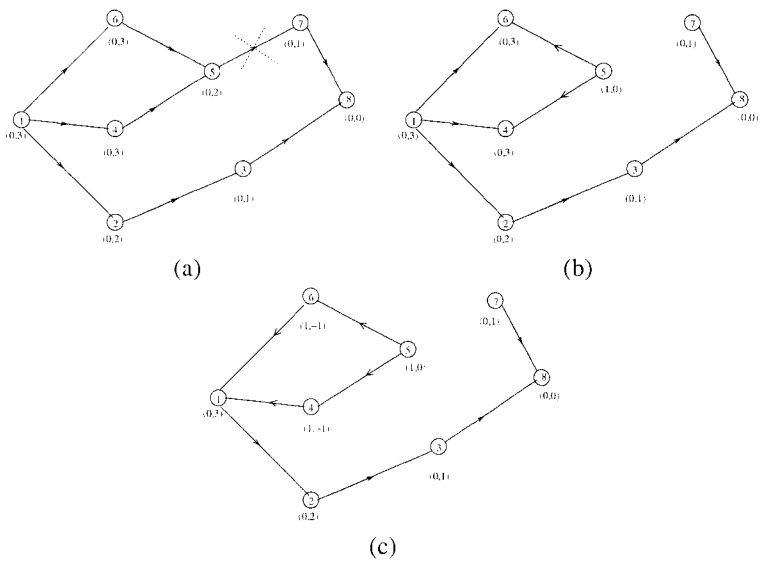
When a node moves, the DAG route is broken and route maintenance is initiated to reestablish the DAG. When the last downstream link of a node fails, that node adjusts its height (by generating a new reference level) to a local maximum. As a result, all links coming into it are "reversed". Then it sends an update message to all of its neighbors reflecting its change in height. A neighboring node left without any downstream links after the change will assign its height to one less than the height



(a) (b)

Figure 2.4: Route Discovery process in TORA (a) Propagation of route request, (b) Propagation of route reply. Numbers in braces are (reference level, height of each node).

mentioned in the update message. This update message will be propagated until it reaches a node having at least one downstream link. If no such node exists, then clearly the link failure has created a partition. This will be ultimately detected in TORA, possibly after some oscillations. Figure 2.5 shows a link reversal and a reconstruction of DAG. TORA floods a *clear packet* through the network to erase invalid routes.



(a) (b)

(c)

Figure 2.5: Link reversal and reconstruction of DAG

On the downside, timing is important in TORA. TORA needs every node to be synchronized via some external time source such as the Global Positioning System. Also, oscillations might occur, especially when multiple nodes detect a partition at the same time, erase old routes, and build new routes depending on each other.

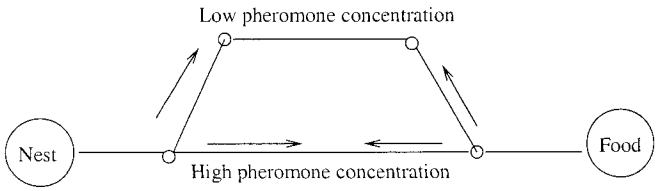


Figure 2.6: Food search behavior of ants.

2.1.2.4 Associativity-Based Routing (ABR)

The fundamental objective of ABR [61] is to find long-lived routes. This is a completely different objective than those of the other approaches. ABR uses the *degree of association* as its metric. All nodes in the network periodically send beacon messages to their neighbors. When a neighbor receives a beacon from a node, it increments the associativity tick corresponding to that node. A larger associativity tick value indicates low mobility of a neighbor which indicates that the path through it will be long-lived with high probability. The route discovery process similar to the other protocols, starts with the broadcast of a BQ message. A node receiving the BQ message appends its associativity tick values with all its neighbors and rebroadcasts. Each successor erases its upstream node's neighbor's tick and only retains the tick value between itself and its upstream node. Each packet arriving at the destination will contain the associativity value of all the nodes along the route. The destination chooses the path with the highest associativity value. If two paths have same value, then the path with the lower hop count is preferred. Once the path is chosen, the destination sends a REPLY packet along the reverse direction of the chosen path. Other than the (complete) route discovery, ABR also accommodates partial route discoveries, invalid route erasures and valid route updates.

2.1.2.5 Ant-colony-based Routing Algorithm (ARA)

ARA [16] is an on-demand ad-hoc routing algorithm. Ant colony algorithms are a subset of *swarm intelligence*. The basic idea of ARA routing is taken from the food search behavior of real ants. On their way to search for food, ants leave their nests and start walking. While walking, ants deposit *pheromone*, a strong chemical substance, which marks their path. The concentration of pheromone indicates a path's usage. The concentration decays over time. When ants reach an intersection they need to randomly choose a branch. So at an intersection they may be divided into different groups. Figure 2.6 shows ants taking two different routes on their way to food. The group of ants taking the shorter path will reach the food first. On their way back home they again choose two different paths. Note that the ants using the shorter path will increase their path's pheromone concentration faster than the other route due to path length. This will lead to all subsequent ants taking the shorter path because the ants choose to follow the paths based on the level of pheromone concentration.

The algorithm proposed in [16] is based on this concept. It initially uses a special small broadcast

packet called FANT (forward ant) and BANT (backward ant) for path establishment. When a FANT or BANT passes through a node it sets a pheromone value on each incoming and outgoing link. After path establishment ARA relies on data packets for path maintenance. Route failures are recovered by special route error packets. Upon receiving a route error message for a certain link, a node deactivates the link by setting its pheromone value to zero.

2.1.2.6 Load aware protocols

There are few load aware routing protocols that can be found in literature. One such protocol is Dynamic Load Aware Routing (DLAR) [27], which is based on AODV and adds load consideration. The load of a node is defined as the number of packets in their buffer at any particular instant. When a node needs a route to a destination it broadcasts a *route request*. Every node receiving the route request, records the node address from which it received the request, establishing a backward path. Then the node attaches its load information and rebroadcasts the route request. The request propagates until the destination receives it. The destination waits for a certain amount of time called *settling time* to receive (almost) all of the requests, then chooses the best path with the least load. Then it uses that route in reverse direction to send back the *route reply*. Unlike AODV, in order to avoid congestion, DLAR prohibits an intermediate node from sending back a route reply. It also uses data packets to carry the most up-to-date load information. In case of a route failure, it issues a *route error* message which is relayed back to the source and erases the route information. The source node then re-initiates a route request.

There exist a few other load sensitive routing schemes, which differ mostly in the way they interpret the "load". Hassanein et al. in [18] proposed the LBAR (Load Balanced Ad hoc Routing) protocol, where the load of a node is defined as the total number of routes passing through the node and its neighbor. Wu and Harms in [68] proposed LSR (Load-Sensitive Routing), where load is defined as the sum of the number of packets buffered at a node and its neighbors.

Both the number of packets buffered in the queues and the total number of routes passing through nodes are questionable definitions of load because they do not consider "contention" as a factor.

2.1.3 Zone Routing Protocol (ZRP) as a hybrid solution example

ZRP [17] is a hybrid protocol that combines the proactive and reactive features. A node in ZRP defines its zone as the set of nodes falling within a fixed number of hops (called 'zone-radius') from itself. Nodes located exactly zone-radius hops away are called the border nodes for that zone. A node may potentially be located in many zones and may be a border node for several zones.

ZRP proactively maintains routing information within local zones and establishes routes on-demand for destinations located in different zones. Interestingly, setting the zone radius to zero will convert ZRP into a purely reactive protocol and setting zone-radius to infinity will convert it into a purely proactive protocol.

If the destination is within the sender's local zone then the route is readily available. Otherwise the sender sends a route request packet, with its own address, to its border nodes. All border nodes search for the destination in their own local zone. If the destination is not found, they append their own address to the route request packet and send it to their border nodes. This process continues until the request reaches a node which has the destination as a member of its zone. Then this node sends back a route reply using the reverse path mentioned in the request packet.

The main advantage of ZRP is that it tends to incur less route discovery overhead (in terms of packets) than a typical "pure-breed" protocol. On the other hand its routing algorithm is more complex than those of purely proactive/reactive protocols.

2.1.4 Position-based routing

Position-based routing algorithms use location information to route packets. They assume that nodes know their geographical position with the aid of special devices such as Global Positioning System (GPS) devices. Each node makes a forwarding decision based on the position information of the destination supplied with the packet by the source, its own position, and the positions of all one-hop neighbors. The main advantage of this protocol is that it does not require the establishment or maintenance of routes. Compass routing, Random Compass Routing, Greedy routing, Greedy Perimeter Stateless Routing (GPSR) are some position-based routing algorithms. A detailed survey of position-based routing may be obtained from [34].

A major drawback of position-based routing is its dependence on location services. Position-based routing algorithms assume that the position of destination can be obtained from location services. Location services are costly and the update of location information of a destination may take a while to propagate in mobile environment, causing packets to be forwarded using less accurate information.

2.2 Medium Access Control (MAC) issues

The sharing of a common transmission medium by multiple nodes is coordinated by a Medium Access Control (MAC) protocol. To a significantly lesser degree than the routing schemes, MAC protocols have also been receiving attention from some researchers. Here are the fundamental issues that MAC protocols need to address in ad-hoc wireless networks [7]:

Collision Avoidance In a wireless environment, it is very difficult to receive data while sending, because, while sending, a large fraction of the signal energy leaks into the received path. This phenomenon is referred to as *self-interference*. This causes a half-duplex mode of operation and collisions can not be detected. As a result, MAC protocols must use collision avoidance principles to try to minimize the probability of collision.

Reliability Wireless links are more error prone than wired links. As a result, some protocols designed for wired environments experience degraded performance in wireless environments. TCP is one classic example. From the MAC layer's perspective, packet losses due to errors can be minimized by incorporating retransmission methods. This can be achieved by introducing acknowledgment (ACK) packets to detect packet errors. If an ACK is not received at the end of transmission, the packet is retransmitted.

Location dependency In free space, the signal strength decays in proportion to the square of the distance. Therefore, detection of a carrier depends on the position of the "transmitter-receiver" pair. Nodes located only within a specific distance from the transmitter can detect the carrier. This raises two interesting issues:

The hidden node problem A hidden node is a node which is within the range of the intended receiver but out of the range of the transmitter. Figure 2.7(a) shows one example. When node A transmits, C cannot detect the transmission. It can falsely assume that the channel is free and start transmitting to B, colliding with A's transmission.

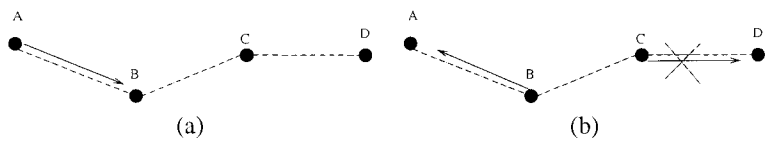


Figure 2.7: (a) The hidden node problem: C is the hidden node, (b) The exposed node problem: C is the exposed node.

The exposed node problem An exposed node is a node which is within the range of the sender but out of the range of the receiver. Consider the same example but with the roles of nodes A and B interchanged. It is shown in Figure 2.7(b). When B transmits to A, node C can hear the transmission and will falsely assume that the channel is busy. But if C has something to send to D it could do so because this transmission will not collide with B's transmission to A.

Medium Access Control algorithms for wireless networks can be broadly classified as *contention-based* and *reservation-based*. In contention-based MAC algorithms, nodes contend for access to the medium. If only one node is contending, then a packet can be transmitted successfully. On the other hand, simultaneous transmission attempts from multiple contending nodes, results in a collision. A contention resolution scheme is used to resolve collisions. Reservation-based schemes are mainly based on Time Division Multiple Access (TDMA) scheduling. Here, time is divided into fixed size slots which are organized into a synchronous frame. Each node is assigned to a unique slot per frame when no one else can transmit, thereby guaranteeing every node a transmission opportunity. Unicast, multicast, and broadcast packets can be easily accommodated this way. In this approach,

the length of the frame depends on a global parameter N, the number of nodes in the network. There also exist some hybrid approaches [59, 37] to tackle this issue which support both point-to-point and multi-point packet transmissions. These approaches initially use TDMA techniques, where the same slot can be accessed by two or more nodes, which resort to a contention-based scheme within the slot [37].

Contention based algorithms are especially suitable for distributed network architecture such as ad-hoc networks. There are several advantages of these algorithms over scheduling-based algorithms. In contention-based schemes, there is no startup overhead. Nodes can instantly start communicating with neighbors without having to wait for any synchronizing events. Another advantage of contention-based MAC schemes is flexibility: they can easily operate in dynamic environments resulting from node mobility, node failure, or energy depletion. In TDMA-based MAC schemes these dynamics may trigger frequent channel re-assignment and thus kill most of the advantages of collision-free operation. Therefore, our focus of interest is distributed contention-based MAC protocols.

The first contention-based MAC protocol for packet radio networks was ALOHA [2]. In that protocol, when a node has a data to transmit, it transmits without listening to the channel. If the transmission collides with another transmission, the node tries again after a random period. The period of time during which a packet transmission may be interfered with is twice the size of packet transmission time. Therefore the maximum channel throughput is very low. To avoid this problem Tobagi et al. [60] proposed another contention-based protocol, Carrier Sense Multiple Access (CSMA). In their design, CSMA addresses the half-duplex nature of wireless communication by having nodes listen for channel activity before deciding to transmit. However, the hidden node problem was not addressed in their design. To address the hidden node problem, Karn [23] proposed a collision avoidance mechanism involving a two-way handshake. In this scheme, a sender node initially transmits a Request-To-Send (RTS) packet. After receiving the RTS, the intended recipient node sends a Clear-To-Send (CTS) packet to the sender. These RTS/CTS packets contain the length of time needed to transmit the packet. Any third party node receiving a RTS/CTS packet refrains itself from sending any packet during the prescribed period described in the RTS/CTS packet. Over the years, many variations and combinations of these two basic techniques have been proposed and, finally a standard named IEEE 802.11 became popular and commercially available [10]. IEEE 802.11 is based on 'carrier-sense' with 'collision avoidance'. According to this protocol, a sender of a packet must first sense the channel to see whether it is free for transmission. If the channel is busy then the sender retries after a random time period. Otherwise, the sender-receiver exchange involves a four-way handshake: RTS/CTS/DATA/ACK. The first two parts (RTS-CTS) avoid hidden nodes and the last two parts (DATA-ACK) increase the reliability of transmission by retransmitting packets in the absence of an ACK. A detailed description can be found in [10].

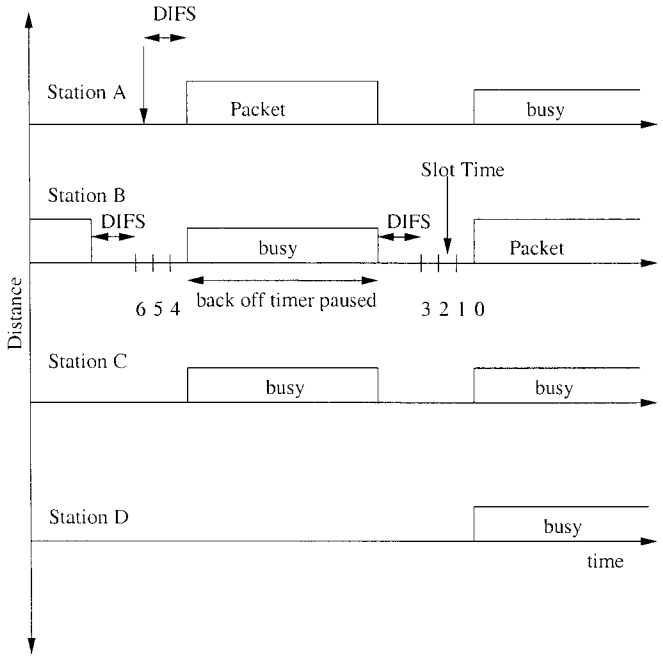


Figure 2.8: Node A broadcasting a packet

Still IEEE 802.11 does not solve all issues. The collision avoidance mechanism described in the protocol is targeted at point-to-point communication. In other words, its effectiveness for multi-point communication is limited and, in particular, the four-way handshake is useless. Multi-point communication involves multiple participants, where, with a single-hop transmission, the packet may be received by more than one interested node. The need for multi-point communication may arise in several situations from an upper layer's perspective. For example, many routing protocols use broadcast transmission services for exchanging neighborhood information and, generally, path setup/recovery for the subsequent point-to-point forwarding. Multicast transmission services involving communities of interests call for forwarding the same packet along a tree interconnecting a subset of all nodes in the network. To support these kind of services, MAC protocols need to provide for reliable multi-point communication.

To explain the problem, we will briefly describe how a single-hop broadcast packet is handled by channel access mechanism within IEEE 802.11 called "Distributed Coordination Function" (DCF). A station with a new broadcast packet to transmit first 'listens' to the channel. If the channel is idle for a certain short amount of time, the DIFS (Distributed Inter Frame Space) interval, then the station can broadcast the packet. All the stations within the communication range of the sending station may receive the packet. On the other hand, if the channel is found to be busy during this interval then, to reduce the probability of collision, the station will defer its transmission for another random "back off" interval before transmitting. The back off procedure first draws a random number in the

range $[0, cw - 1]$, where cw is called the "contention window" and it varies based on the number of transmission attempts for the current packet. Then the station divides the time immediately following an idle DIFS into equal size slots and starts sensing them. If it finds an idle slot, the back off counter is decremented by 1. If the channel is busy at any time during the slot then the back off procedure is suspended and resumes once the station senses the channel idle for a DIFS period. As soon as the counter reaches zero, the transmission may start.

Figure 2.8 shows an example of a broadcast packet transmission. In this example, it is assumed that stations B and C are within the communication range of station A, and station D is within the communication ranges of stations B and C but not A. At the end of a transmission, B wants to start a new broadcast. It waits for a DIFS period and chooses a random back off value of 6. B senses 3 consecutive idle slots and decrements its back off counter, but, after that, the channel becomes temporarily busy due to a packet transmission by A. B's back off procedure is paused during A's packet transmission. At the end of A's transmission, B senses that the medium is free for a DIFS period and resumes decrementing its back off counter which reaches zero after another 3 slots. B starts broadcasting. Here we assume that station C and D had no packets to transmit and were just listening to the broadcasts of stations A and B.

Note that the multi-point communication is handled without any RTS/CTS handshakes. Even ACK packets are not sent at the end. Therefore the multi-point communications are much more unreliable than point-to-point communications. No solution to solve these issues in contention-based protocols has been proposed to date.

2.3 Power control for energy efficiency

The power control problem can be defined as choosing an optimal power level (not exceeding the maximum possible power level) for each node while optimizing some cost metrics and satisfying some constraints. Note that, by choosing different power levels, nodes can automatically modify the topology of the whole network and create topologies that hold some desirable properties. This set of desirable properties includes:

(1) *Connectivity:* the most desirable property of topology control is that the modified topology remains connected provided the original one was. More precisely, if there exists a path between a pair of nodes in the original maximum-powered network, then there should also exist a path between the same pair in the modified topology.

(2) *Energy-Efficiency:* the topology control algorithm does not solve the routing problem, but any routing protocol running on top of it will use only those paths that are provided by the modified topology. Thus it is also desirable that the modified topology preserves all the energy-efficient paths between all pairs of nodes. One metric to measure energy-efficiency is the *energy stretch factor* [49]. $C_G(u, v)$ denotes the energy needed in the minimum-energy path of maximum-

powered graph G. Similarly, $C_T(u,v)$ denotes the energy needed in the minimum-energy path in the modified topology graph T. Then *energy stretch factor* is defined as

$$\text{Energy Stretch Factor} = \max_{u,v \in V} \frac{C_T(u,v)}{C_G(u,v)}$$

It is desirable that the *energy stretch factor* is 1 or close to 1.

(3) *Throughput:* the reduction of power level may result in many concurrent transmissions taking place, which may not be possible in the maximum powered network due to interference. Transmissions with high power increase the range of interference and reduce the traffic-carrying capacity of the network. One goal of topology control is to accommodate as much traffic as possible (i.e. maximizing the capacity of the network).

(4) *Planar Spanner:* another desirable property may include forcing the topology to be planar. Many routing algorithms depend on a planar topology to guarantee message delivery. Greedy Face Routing (GFR) [5] and Greedy Perimeter Stateless Routing (GPSR) [24] are examples of such protocols. Another desired property of a modified topology is that the distance between any two nodes should be within a constant factor (spanner).

(5) *The number of edges and maximum node degree:* other useful goals are to reduce the number of edges as much as possible, subject to the connectivity constraint and ensuring that the maximum node degree is bounded. A small node degree indicates lower MAC-level contention, reduced interference, and a smaller number of hidden and exposed terminals. Low node degree is a useful criterion. Bluetooth-based ad-hoc networks prefer that the master node degree be less than 7.

Our primary interest is controlling power to achieve energy-efficiency. Why do we need to control power? Simply because it affects the performance of many layers in the protocol stack. It determines the range of transmission affecting network layer in terms of the choice of intermediate nodes to route the packets. Transmission power determines the amount of energy consumed to send the packets. It also affects contention. The number of nodes contending for the channel implicitly depends on the range of transmission (affecting MAC layer performance). Power also determines the level of congestion affecting transport layer performance. It affects the traffic carrying capacity of the network which affects overall network performance. The power control problem is a classical cross-layer design problem affecting many layers. It is always important to determine the correct power level of each individual node in the network.

The following issues need careful consideration before designing a good power control algorithm. Any such algorithm

(1) Should be entirely distributed. Dependence on any global information may become counter-productive.

(2) Should generate only bidirectional links.

(3) Should not be very vulnerable to mobility. The effect of mobility should be purely local, i.e., affecting only a few nodes located in the vicinity of changes.

(4) Should preserve the connectivity of the network.

Now we provide a brief survey of power control algorithms. Many researchers have been contributing to this area within the last few years demonstrating different power control algorithms. Roughly, there have been three distinct approaches.

The first approach tried to solve the power control problem in the MAC layer. The approach essentially modifies the MAC layer. Monks *et al.* [35] proposed a modification of IEEE 802.11's RTS-CTS handshake procedure. They argued that if a third node can estimate the distance of the recipient from the signal strength of CTS packets, then it may continue transmitting at low power as long as it does not interfere with the receiving node. This will eventually increase the throughput. However they did not make any attempt to reduce power consumption of broadcast packets. Sing *et al.* [53] proposed powering-off the transceivers to reduce energy consumption when they are not actively transmitting or receiving any packet. The second approach is the so-called "power-aware routing." Most of the schemes use the distributed Bellman-Ford algorithm with power as the cost metric. Some metrics mentioned in [54] are: energy consumed per packet, time to network partition, variance in node power levels, cost per packet, etc. A detailed survey of power-aware routing can be found in [70]. The third approach is what we are mostly interested in. This approach solves the power control problem in the routing layer but does not solve the routing problem. In other words this is not an integrated approach of routing and topology control. Rather power-aware routing protocols can be used in conjunction with this approach. The algorithms that use this approach vary based on the different goals they attempt to achieve subject to different constraints. Our focus is upon those algorithms that are based on a completely distributed approach (thus being compatible with the ad-hoc paradigm), lead to power efficient operations, and preserve connectivity. Now we will discuss some of those approaches in brief.

2.3.1 CONNECT and its extension

Ramanathan et al. [50] described two centralized algorithms to minimize the maximum transmission power used by any node while maintaining connectivity (named CONNECT) or biconnectivity (named BICONN-AUGMENT). CONNECT is a greedy algorithm that starts by assigning each node to a separate component. Then the algorithm iteratively merges connected components until there is just one. Node pairs are selected in the order of increasing mutual distance. For any particular pair, if they belong to different components, then the transmission power of each node is increased until they just reach each other. BICONN-AUGMENT is an augmentation run after the CONNECT

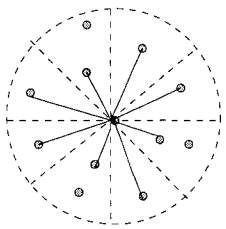


Figure 2.9: Yao Graph

phase to extend the connected topology into a biconnected one. Moreover, a post-processing phase is performed to ensure a per-node minimal power assignment by deleting redundant edges.

Two distributed heuristics called Local Information No topology (LINT) and Local Information Link-State Topology (LILT) were introduced to work in mobile environment. The major differences between these two heuristics are in the underlying information used and the desired property to maintain. In LINT, each node is supplied with three parameters related to node degree: a desired node degree d_d, an upper limit on node degree d_h and a lower limit on node degree d_l. Every node periodically checks the node degree constructed from the neighbor table and adjusts the transmission power to keep the node degree within the threshold limits. LILT is an improvement over LINT, which overrides the high limit of node degree if topology changes cause undesirable connectivity.

The major drawbacks of CONNECT and BICONNECT is that they are centralized algorithms requiring global information. Collecting global information may become counter-productive and is difficult in mobile environments. In addition, neither LINT nor LILT guarantees network connectivity.

2.3.2 Cone-Based Topology Control (CBTC)

This approach is based on a *Yao Graph* [69], defined as follows: Consider a circle around a node u with the radius defined by the transmission power. This circular region is divided by k equally spaced line segments originating at u ($k \geq 6$). Each segment is called a 'cone'. In each cone we choose the shortest edge uv, and add a directed link $\overrightarrow{uv}$. The resulting graph is called Yao Graph. See Figure 2.9 for an example.

Li *et al.* [29] proposed a topology control algorithm which eventually generates a graph structure similar to Yao structure. The basic CBTC algorithm takes a parameter α. Each node u determines a power level $p_{u,\alpha}$ such that in every cone of degree α surrounding u there is at least one node reachable by $p_{u,\alpha}$. Each node starts with a initial small power level and gradually increases the power until the above condition is satisfied. Then the graph G_α contains all edges uv for each node v that was found in its search process. Li et al. also proved that if $\alpha \leq \frac{5\pi}{6}$ the resultant graph is connected provided the original one was. The algorithm can run periodically to reconfigure the topology under mobile conditions.

Several optimizations have been proposed to run on top of this basic algorithm. A *shrink-back* operation may be included at the end so that boundary nodes can broadcast with less power. Also, if $\alpha \leq \frac{2\pi}{3}$, then asymmetric edges can be removed while preserving connectivity.

One of the major drawbacks of the algorithm is the determination of suitable initial power level and the increment of power level at each step. Li et al. did not provide any definitive algorithm for determining those parameters. In a highly dynamic environment with varying node density, the choice of these two parameters may have a significant impact on the number of overhead messages generated while creating the desired topology.

2.3.3 COMPOW

Narayanaswamy *et al.* [38] proposed a power control protocol named COMPOW. Their approach is to choose the smallest common power level by each node while 1) maintaining connectivity, 2) maximizing traffic carrying capacity, 3) reducing contention at the MAC layer and 4) requiring low power to route the packets. In their approach, they first choose a set of power levels available for adjustments on a particular node. Each node runs several routing daemons in parallel, one for each power level. Each routing daemon exchanges control messages at the specified power level to maintain its own routing table. The entries in different routing tables are than compared by each node to determine the smallest common power that ensures the maximum number of connected nodes in the network. In particular, if $N(P_i)$ denotes the number of nodes that are connected at power level P_i, then the required power level is set to the smallest power level P_i for which $N(P_i) = N(P_{max})$. One advantage of this approach is that it automatically generates symmetric links.

The major drawback of this approach is its impractical assumption of homogeneous distribution of nodes. If the nodes are distributed in a non-homogeneous fashion, even a single node located far apart from a group of nodes that are close to each other may cause every node to choose a higher power level to ensure connectivity. Thus a single node may affect the performance of a closely located group of nodes. Also each node runs multiple daemons which need to exchange link state information with that node's counterpart, causing a significant message overhead.

2.3.4 CLUSTERPOW

CLUSTERPOW [25] was designed to overcome the shortcomings of COMPOW [38]. COMPOW is a special case of CLUSTERPOW. It was designed to work on non-homogeneous deployment of nodes. Closely located nodes can form a cluster and choose a small common power to interact with each other. Different clusters will communicate with each other at a different high power level. Thus most of the intra-cluster communication is done at lower transmit power level and inter-cluster communication is done using a higher power level. The formation of clusters is implicit. There are no explicit cluster heads or gateway nodes.

As with COMPOW, in this approach, each node runs multiple daemons which need to exchange

link state information with that node's counterpart, causing a significant message overhead.

2.3.5 LMST: an algorithm based on the Minimum Spanning Tree

N. Li *et al.* [30] proposed a distributed topology control algorithm based on minimum spanning tree construction. Their algorithm achieves 3 goals: 1) connectivity, 2) bounded node degree (≤ 6) and 3) provision of bi-directional links. The algorithm works as follows. Each node collects the position information of its neighbors reachable with the maximum power. Based on this information, a node u creates its own local minimum spanning tree among the set of neighbors where the weights of the edges are the necessary transmission power levels between two nodes. Once the tree construction is done, node u keeps those nodes on its tree that are one-hop away as its neighbors in the final topology.

One major drawback of LMST is that it does not preserve the minimum-energy paths between nodes in the final topology.

2.3.6 A relay-region based approach

Rodoplu *et al.* [52] proposed the notion of a *relay region* based on a specific power model. With respect to a node pair (i, r), a node j is said to fall under relay region of r if transmitting via r consumes less power than transmitting directly to j from i. The *enclosure* of a node i is the intersection of the complements of all relay regions of nodes that are reachable from i with maximum power. Node i only needs to maintain connectivity with those nodes that do not fall in the relay region of any other node. To deal with mobility, each node periodically gathers its neighbor's position information and reconstructs the enclosure. The resulting topology is guaranteed to preserve minimum-energy paths.

One major drawback of this approach is the use of a specific (restrictive) power model.

2.3.7 Small Minimum-Energy Communication Network (SMECN)

The work that most closely relates to our work is SMECN [28] by Li *et al.*. This work is a modification of the relay-region based approach [52]. The first modification is in the use of maximum power to gather neighbor information. Instead of using maximum transmission power, a node will start with relatively low power and will gradually increase the power until the *enclosure* is found or the node has reached the maximum power. This approach generates fewer edges than the approach used by Rodoplu *et al.*. Also this approach is guaranteed to preserve minimum-energy paths between all pairs of nodes. More discussion about this approach is presented later in this thesis.

One major drawback of this approach is in the choice of initial power and the amount of the power increment at each step. There are no specific guidelines on how to choose those two parameters optimally.

A few other studies in this area are worth mentioning. Wattenhofer *et al.* [63] also proposed a two-phased algorithm that consists in creating a variation of the Yao graph followed by a Gabriel Graph. The Gabriel Graph [15] contains an edge uv from the original graph G if and only if the circle having edge uv as diameter does not contain any other vertex from G inside. The combined structure of Yao graph and Gabriel graph was shown to be more sparse while still having a constant bound on the *energy stretch factor*. Huang et al. [19] proposed a topology control algorithm which utilizes directional antennas.

2.4 Broadcasting

The task of broadcasting, as an application-level functionality, is to convey a message from a single source to all other nodes in the network. Such a task has many applications in ad hoc wireless networks. Reactive routing algorithms use broadcasting to discover a route from a source to a destination. Proactive routing protocols use broadcasting for sending periodic updates of cost metrics from every node within the network. Location-aware routing protocols may use broadcasting to discover the position of a destination. Also broadcasting can be used for paging a particular host or for sending an alarm signal to the entire network. In sensor networks, broadcasting is used for collecting and disseminating information such as temperature, air pressure or noise level.

The simplest way to broadcast a packet is to use *flooding*. With flooding, every broadcast packet is transmitted exactly once by every node in the network. The broadcast propagates in layers outwards from the source and terminates only when every node has received and transmitted exactly once a copy of the original packet. This process is wasteful because, in most circumstances, the same packet can reach all nodes with a considerably smaller number of retransmissions. The following issues need thoughtful consideration before designing a broadcasting algorithm. Any such algorithm:

(1) Should be entirely distributed and use as much local information as possible. Dependence on any global information may become bandwidth-intensive.

(2) Should reduce broadcast overhead (redundant retransmissions) as much as possible.

(3) Should be reliable and guarantee that every node in the network receives the broadcast packet.

(4) Should minimize amount of time (latency) needed to complete the operation. The latency of broadcasting is defined as the difference between the moment when a broadcast is initiated by a source and the moment when the last node in the network receives a copy of the broadcast packet. A good broadcasting algorithm should incur low-latency.

Broadcast optimization techniques in the wireless environment have been the subject of several studies in the past few years [39, 58, 32, 42, 46, 33]. In this dissertation, we classify them into

two categories: (1) unreliable, and (2) reliable. Reliability is defined as the ability of a broadcast protocol to reach all the nodes in the network. Note that a reliable broadcast algorithm may become unreliable due to imperfection in the underlying MAC protocol. Assuming an ideal MAC layer, we will call a broadcast protocol *reliable* if it guarantees that a broadcast message will reach every node in the network.

2.4.1 Unreliable broadcast protocols

Unreliable broadcast protocols are mainly based on flooding. Several attempts have been made to make flooding efficient. The first category of broadcasting protocol attempts to eliminate redundant broadcasts as much as possible, while at the same time, achieving high reachability. But these protocols do not guarantee 100% reachability.

With probabilistic flooding [39], a node receiving a broadcast packet retransmits it with a certain probability $P < 1$. The idea is that in a reasonably dense network, the statistical likelihood of reaching all nodes may still be high, even if only some of them (selected at random) rebroadcast the packet.

In counter-based schemes [39], a node receiving a broadcast packet waits for a certain (random) amount of time before deciding whether the packet should be retransmitted. If while waiting, the node receives a predetermined number of copies of the same packet, then it will not retransmit itself—concluding that most likely all its neighbors have already received the packet as well. Other techniques involving thresholds, also considered in [39], include distance-based and location-based schemes. In distance-based approach, the packet is not retransmitted if during the waiting period, the node receives another copy of the packet transmitted by a neighbor located not further than some predetermined distance d from the node. In location-based approach, a packet is retransmitted if the additional area that can be covered by the node's rebroadcast is greater than some predetermined threshold A from the node.

The major drawback of all of these algorithms is the lack of guarantees that a broadcast packet will, in fact, reach all nodes (even assuming that the network is connected). Also, their performance is extremely sensitive to the threshold values, whose selection is not obvious a priori (see [65] for a detailed performance study).

The waiting time before retransmitting a broadcast packet can be chosen in a way that accounts for different coverage by different nodes. For example, [58] suggests a distance-dependent waiting time whereby a node receiving multiple copies of a broadcast packet calculates the area covered by them. At the end of the waiting period, the node knows whether its own transmission would cover any new area and does not transmit if this is not the case. This approach assumes location-awareness, i.e., nodes know their own location (e.g., via GPS) and keep track of the location of their neighbors.

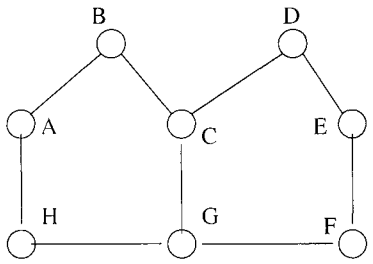


Figure 2.10: Connected Dominating Set

2.4.2 Reliable broadcast protocols

In reliable broadcast protocol, a broadcast packet is guaranteed to reach every node in the network. The set of nodes that participate in relaying is called the *connected dominating set*. A dominating set is the set of nodes $\mathcal{S}$ such that for any node u either $u \in \mathcal{S}$, or there is a node $v \in \mathcal{S}$ such that u and v are neighbors. To achieve 100% reachability, only the nodes in a connected dominating set are required to retransmit. Therefore, the optimal broadcast problem can be solved by finding a connected dominating set of minimum size. In Figure 2.10, some possible connected dominating sets are $\{A, C, F, G, H\}$, $\{A, B, C, D, E\}$ etc. The minimum connected dominating set is $\{B, C, D, G\}$. Once the MCDS is constructed, broadcasting becomes simple. For a given graph the following two cases may arise for broadcasting:

(1) The source is included within the MCDS. In this case, the number of packet transmissions needed to complete the broadcast is equal to the size of the MCDS.

(2) The source is not included in the MCDS. In this case the number of packet transmissions is 1 plus the size of the MCDS.

Unfortunately, finding a *minimum connected dominating set* (MCDS) is an NP-complete problem [31]. Thus approximation algorithms must be used for non-trivial cases. If global network topology information is available, then one of the efficient approximation algorithms for the MCDS problem, like the algorithm proposed in [11], can be used. This algorithm starts by coloring all the nodes as white. Then the node with the maximum node degree is chosen, colored black and all its neighbors are colored grey. In the next step, the grey node with the maximum number of white nodes as neighbors is selected, marked as black, and all its neighbors are marked as grey. The process continues until no white nodes exist. The set of black nodes comprise the MCDS. One major drawback of this algorithm is its dependence on global topology information. In an ad hoc wireless network, topology may change frequently due to node mobility, the environment, or many other factors. Therefore, many heuristic algorithms that can compute connected dominating sets in a distributed fashion based on local information only, have been proposed. All distributed heuristics seek a small forwarding set with the least-possible overhead.

Lim and Kim proposed two heuristic algorithms in [32]. With one of them, dubbed *self-pruning*,

a node retransmitting a broadcast packet includes the list of its neighbors in the packet header. A receiving node compares its own neighbor list with the one in the header. If its own neighbor list is a subset of the list in the header, then all its neighbors must have received a copy of the packet and the node can refrain from forwarding it. This way, the node is able to determine whether its neighbors have been covered by the received copies of the broadcast packet without knowing their exact locations. Wu and Dai [66] provide a generalized view on self-pruning by considering information about k-hop neighbors (instead of one or two). It seems that any scheme considering more than two hops is not very practical in the face of topology changes implied by mobility. The overhead of collecting and updating the information about 3- or 4-hop neighbors will most likely nullify any savings on retransmissions.

The second algorithm proposed in [32] and called *dominant-pruning* exploits 2-hop neighbor information. In dominant pruning, the sending node proactively selects adjacent nodes that should retransmit the packet to complete the broadcast. Each node maintains a subset of its one-hop neighbors (called *a forward-list*) whose retransmissions will cover all nodes located two hops away from the node. Finding such a minimum size set is NP-complete [32]. Instead Lim and Kim used a greedy set cover algorithm to construct such a set. It is important to know how a node creates its forwarding list. Let us assume forwarding list F is initially empty. The algorithm starts with another list of uncovered nodes which are exactly 2-hops away and not covered by the previous transmission. Then the node includes a neighbor in the set F which can cover the maximum number of nodes that are yet uncovered. Once that adjacent node is added to F, all its neighbors are removed from the list of uncovered nodes and the process is repeated until all uncovered nodes are excluded from the list. In the header of every broadcast packet, the sender includes this forwarding list F. If an adjacent node is included in the forwarding list then it relays the packet, otherwise not. The process is iterated until the broadcast is complete.

A similar algorithm, known as *multi-point relaying*, was introduced in [46]. The major difference between *dominant-pruning* and *multi-point relaying* lies in the way the forwarding list is created. In *multi-point relaying*, the forwarding list is called the "MPR" (Multi Point Relay) set. A node includes a neighbor in its MPR set only if it is the only neighbor on the route to a two hop neighbor. The node iteratively includes all such neighbors in the MPR set. Then it picks a neighbor that is not already included in the MPR set and that covers the most nodes that are uncovered by any member of the current MPR set. The last step continues until all 2-hop neighbors have been covered by some member of the MPR set.

Peng et al. [42] present a modification of the self-pruning algorithm named SBA (for *Scalable Broadcast Algorithm*). Instead of taking a rebroadcasting decision immediately after receiving a broadcast packet, a node postpones its decision until a predetermined *waiting time* expires. Similar to [32], a node does not rebroadcast if the duplicate packets received during the waiting time cover all its neighbors. The algorithm uses 2-hop neighbor information to determine the redundancy of

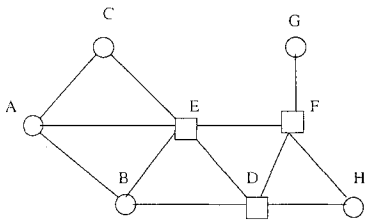


Figure 2.11: Intermediate, inter-gateway and gateway nodes

the broadcast packet. The performance of this scheme appears to be very sensitive to the length of the waiting period.

The dominant-pruning algorithm proposed by Lim and Kim in [32] was later enhanced by Lou and Wu [33]. Two new approximations were proposed: *Total Dominant Pruning* (TDP) and *Partial Dominant Pruning* (PDP). Both algorithms incur fewer retransmissions than the original scheme. Total Dominant Pruning requires the sender to piggyback onto each broadcast packet information about all its one- and two-hop neighbors. With this information, any receiver node can prune the sender's all 1-hop and 2-hop neighbors from its set of nodes that need to be covered. Partial Dominant Pruning relaxes the requirement of the piggybacking technique at the cost of worsened performance. In PDP, the receiver directly extracts the neighbors that are common to both sender and receiver and prunes their 1-hop neighbors from its set of nodes to be covered.

There exist some other mentionable studies based on dominant pruning. Wu and Lou [67] extended this method to the cluster network. The main advantage of using clustered network is to limit the worst case size of forwarding sets. In fact, they have shown that the cluster graph approach converts any dense graph to a sparse graph with guaranteed constant approximation ratio. [1] Sisodia et. al. [55] introduced the notion of "stability" in selecting the forwarding node set. They proposed a weight function that considers the temporal and spatial stability of a node's neighbor when creating the forward node set. A description of some other techniques can be found in [57].

The dominant-pruning-based method and its variants construct the connected dominating set starting from the source node. Therefore the forwarding node set changes if we have a different source. Now, we will discuss a few broadcast algorithms that construct a fixed forwarding node set.

The algorithms for constructing fixed forwarding node sets partition the set of all nodes into internal and external nodes, and use only the internal nodes to forward broadcast packets. Wu and Li [21] proposed a simple scheme for constructing the set of internal nodes. A node is called an *intermediate* node if it has two neighbors that are not neighbors to each other. In Figure 2.11, all nodes except node C, G and H are intermediate nodes. Next, Wu and Li reduced the number of intermediate nodes by introducing the notion of *inter-gateway* nodes. If there exist two nodes u and

[1] the worst case ratio of selected forward set size with respect to the minimum connected dominating set size.

v such that the neighbors of node v are also neighbors of node u and $id(v) < id(u)$, then we can say that node v is "covered" by node u. Note that all the neighbors of node v will automatically receive the broadcast via forwarding from node u. All non-covered *intermediate* nodes are called *inter-gateway* nodes. Again in Figure 2.11, nodes A and B are covered by node E and hence are excluded from the set of inter-gateway nodes. All inter-gateway nodes are shown as rectangles. Next, the set of inter-gateway nodes can be further reduced by defining "gateway" nodes as follows: suppose there exist three inter-gateway nodes u, v and w such that a neighbor of v is also a neighbor of either u or w, and the *id* of node v is lowest among the three. In this case, node v is not a *gateway* node. In Figure 2.11, node D is not a gateway node. So the set of gateway nodes is $\{E, F\}$ which is also the final connected dominating set. Stojmenovic [56] modified the algorithm to introduce location awareness. There also exists a class of power-aware broadcast algorithms [20], which are beyond the scope of this thesis.

Chapter 3

The Tiny Ad-hoc Routing Protocol (TARP)

3.1 Introduction

Routing in ad-hoc wireless networks is the act of moving information from a source to an arbitrary destination across the network. Traditional routing involves two basic activities: determining optimal routing paths and transporting information in groups (called packets) through the network. Regardless of the high-level paradigm and the manner in which the routing information is acquired, most protocols try to identify the routes before forwarding packets. In source routing schemes, this means a precise identification of every single hop to be made by the packet before the packet departs the source node. In hop-by-hop schemes, each node knows the identity of the next node along the packet's path from source to destination.

Despite the fact that the wireless environment is inherently multicast, this *free feature* is rarely exploited during the actual forwarding of session packets, although all protocols necessarily take advantage of broadcast transmission during various stages of route discovery, when the configuration of available routes or neighboring nodes is unknown or uncertain. Also, in many protocols, e.g., DSR [22] and AODV [43], a node is allowed to overhear (and cache) routing information exchanged by other nodes, which can be viewed as a form of turning the broadcast nature of the medium to the protocol's advantage.

However, once the route has been established, the forwarding of session packets is carried out using point-to-point transmission. At first sight this is obvious: the whole idea of establishing a route is to find out which nodes should be responsible for forwarding. Once this knowledge has been acquired, it only makes sense to send the packets over the established path.

It is usually assumed that the cost of information exchange during route discovery, when the knowledge of the requisite elements of network configuration is imperfect, is higher than the cost of point-to-point forwarding after that knowledge has been acquired. But formally, from the viewpoint of raw bandwidth usage of the wireless channel, it makes no difference whether a transmitted packet

is addressed to (and intended for) one specific neighbor, or whether it targets all/any neighbors that can hear it.

Of course, the assumption about the poorer performance of broadcasting compared to point-to-point transmission is not unwarranted. First, the family of collision avoidance schemes for wireless channels based on IEEE 802.11 [13, 64, 4], significantly favors point-to-point transmission with respect to the reliability of channel acquisition and effective bandwidth utilization. Second, from the viewpoint of reliability of the data-link layer, a point-to-point transmission can be easily acknowledged (with a special provision for acknowledgments in the IEEE802.11 MAC layer [4]), which is more than can be said about a broadcast transmission with no clearly defined single recipient. Third, one of the primary objectives of an ad-hoc routing protocol is to minimize resource usage. In this context, it is natural to restrict the path traveled by session packets to a well-defined (optimum) sequence of nodes. In other words, there is no reason to broadcast if the naturally preferred approach is to learn the identity of the recipient first.

In this research, we show that the operations of route discovery and forwarding can be combined and made indistinguishable, and the result of this amalgamation, in terms of end-to-end performance, need not be inferior to other ad-hoc routing schemes, e.g., based on point-to-point forwarding. We introduce an ad-hoc routing scheme in which forwarding is inherently broadcast-based in that a transmitting node never cares about the identity of its next-hop neighbor. What only matters, is the identity of the source and destination, i.e., packets are addressed exclusively within the transport layer. The proposed scheme is a refinement of straightforward flooding assisted with several heuristics that reduce its range to a narrow stripe of nodes along the shortest path between source and destination. The width of this stripe is adjustable with static and dynamic parameters that account for the expected or perceived mobility patterns, node density, and the required quality of service (QoS).

The performance of our proposed scheme in terms of flooding confinement and convergence (i.e., global resource usage) depends on the amount of local resources (memory and processing speed) available at individual nodes. Notably, with a smaller amount of resources, the protocol still operates correctly, although it may yield suboptimal paths and slower convergence. In particular, owing to its essential simplicity, our approach can be used in very inexpensive low-bandwidth devices (smart cards, sensors) as well as in complex and high-performance systems.

We also suggest an enhancement to the IEEE 802.11 collision avoidance scheme to improve its performance for the kind of multicast packet exchange needed in our protocol. Notably, besides increasing the reliability of forwarding, this new feature can also be used as a basis for new heuristics that facilitate path convergence and reduce global resource usage.

3.2 The protocol

The protocol presented in this section was conceived [40] as a proprietary solution of Olsonet Communications[1] and intended for small-footprint wireless devices organized into small to moderately sized ad-hoc networks, e.g., within the area of a golf course, shopping mall, or campus. The underlying assumptions were automatic configurability of nodes, maintenance-free operation, small to trivial memory requirements at a node, low power, and completely distributed operation. The protocol was implemented under PicOS [3] on eCOG-based[2] and MSP430-based[3] microcontroller boards.

3.2.1 An overview

According to the simple idea of flooding, a node willing to send a packet to some destination simply broadcasts it to the neighboring nodes. A node receiving a packet checks its destination address. If the node is the intended recipient of the packet, the packet has reached the end of its path (it is received and passed to the Transport Layer). Otherwise, the node may decide to re-broadcast the packet.

To describe TARP we have to explain the meaning of the word "may" in the previous sentence. Clearly, the node should not forward blindly all received packets that happen to be addressed elsewhere. Even a naive flooding protocol must take measures to limit the range of flooding. Among the simplest of those measures is restricting the number of hops that a single packet is allowed to travel. In addition to this obvious idea, TARP implements three more techniques (called rules) that heuristically limit the number of stray packets wandering in the network. The exact behavior of those rules is governed by a set of parameters that determine their focus (or fuzziness). For illustration, the SPD (Suboptimal Path Discard) rule acts to restrict all traffic between a pair of nodes to the proximity of the shortest path connecting them. Depending on its focus, the rule may allow some fuzziness by exploring a few (alternative) paths at the same time. While this approach uses more network resources, it provides for a better responsiveness to the dynamically changing configuration of intermediate nodes. The proper identification and implementation of this trade off is what ad-hoc routing is mostly about.

The rules of TARP can be viewed as a multi-part algorithm for determining whether a received packet that does not happen to be addressed to the current node should be retransmitted (forwarded) or dropped. The rules are executed in sequence, and the first one that says that the packet should be dropped terminates the execution of the chain. Thus, the rules are restrictive in nature: their role is to control (limit) the (otherwise unrestricted) flooding.

Once the packet has passed through all the rules, and none of them has decided that the packet

[1] See http://www.olsonet.com/
[2] See http://www.cyantechnology.com/ .
[3] See http://www.msp430.com/ .

should be dropped, the node will queue the packet for forwarding. We say that such a node is *eligible* to push the packet one hop forward on its way to the destination.

The concept of eligibility defines the major criterion of progress in TARP. The responsibility of a transmitting (forwarding) node is to pass the packet to at least one eligible neighbor. The node assumes that its forwarding task has been accomplished when it can tell with reasonable confidence that at least one eligible neighbor has successfully picked up the packet. For the sake of completeness of the progress criterion, we assume that the destination itself is also eligible, i.e., it provides the same kind of feedback to the neighbors as a forwarding node.

3.2.2 Packet header

Packets in TARP are forwarded simply by being retransmitted. While there is no need to modify the addressing information in the headers of forwarded packets, some header information gets updated at every hop. In addition to the obvious source/destination address pair $<S, D>$ (belonging to the transport layer), the TARP-specific (network layer) header components include:

- s—the session identifier unique for a given $<S, D>$ pair,
- n—the sequence number of the packet within its session,
- k—the retransmission count of the packet,
- r—the maximum length of the path that the packet is allowed to travel expressed as the number of hops,
- h_f—the number of hops traveled by the packet so far,
- h_b—the total number of hops traveled by the last packet on the reverse path from D to S,
- m_f—the mobility factor on the forward path from S to D,
- m_b—the mobility factor on the reverse path from D to S.

The sizes of those fields may depend on the application, but most of them can be very short. For example, s and k may use 4 bits each, n, r, h_f, h_b, may each fit into 5 bits (note that the range of packet sequence number depends on the ARQ scheme used by the transport layer and shouldn't be too large in a wireless environment), and both m_f and m_b may be stored into 2-bit fields (representing one of four quantized values). This yields 32 bits of the TARP-specific components, with the packet sequence number being in fact shared with the transport layer.

The role of all those fields will be explained in the next section. The tuple $<S, D, s, n, k>$ is called the packet *signature*. It uniquely identifies a single packet within a certain time frame. The mobility factor is a parameter (passed by the respective end-point of the session) that hints at the desired aggressiveness of the rules in their effort to eliminate the redundancy of paths. Note that in

contrast to the traditional approach to implementing a hop number limit, whereby the remaining hop count of a packet is decremented toward zero, TARP uses two fields: the bound set by the source remains constant, while a separate field stores the increasing number of hops traveled by the packet. This is because both values are needed by the rules.

3.2.3 The first rule

The first rule, called DD for Duplicate Discard, determines whether a packet with the same signature has been recently forwarded by the node. The rule uses a cache of signatures recycled in a First-In-First-Out (FIFO) manner with an additional timing out of old entries—to avoid a wrap-around of n. The simple formula used by TARP to determine the amount of time after which a signature should expire from the DD cache is:

$$T_r = F_c \times t_{avg} \times (r - h)$$

where, t_{avg} is the average transmission time (see below), and F_c is a parameter called the flooding constant. Note that T_r is proportional to the packet's expected distance from the destination. The signature of a packet that is near the end of its trip is not likely to be needed for very long. Also packet duplicates are less harmful at a node located close to the destination than far away from it.

While the premature removal of a DD cache entry may affect the efficiency of TARP (some packet duplicates may pass undetected and be unnecessarily forwarded), it does not affect the formal correctness of the scheme. Thus, the amount of memory allocated to the DD cache is flexible: in some applications it can be minimal—at the cost of increased flooding and suboptimal routing performance.

TARP estimates the average transmission time t_{avg} formally defined as the interval elapsing from the moment a packet becomes queued for transmission (the node considers itself eligible to forward the packet), until the node concludes that the packet has been passed to at least one eligible neighbor. The role of t_{avg} is to estimate the expected packet processing time (the cost of making a hop through the node) in a way that accounts for dynamic conditions affecting this cost, e.g., local congestion. For calculating t_{avg}, TARP samples the processing time of individual packets passing through the node. This is accomplished with a simple trick that requires a single signature buffer, regardless of the packet queue size.[4] The calculated value is an exponential moving average of the samples, i.e., it is updated as follows:

$$t_{avg} = C_a \times t_{avg} + (1 - C_a) \times t_t$$

where t_t is the last sample, and C_a is a constant whose value lies between 0 and 1 (typically 0.65). Note that, according to this equation, the importance of past samples decreases gradually and increases the importance of the recent ones. A larger value of C_a (close to 1) puts more emphasis on the history while a smaller value (close to 0) favors recent measurements.

[4]What we mean here is the data-link layer queue. TARP does not queue packets in the network-layer

3.2.4 The second rule

The second rule is called SPD for Suboptimal Path Discard. Its objective is to avoid forwarding a packet via a route that takes it too far from the shortest path between source and destination. The rule uses its own cache (the SPD cache) storing tuples $<S, D, h_{DK}, h_{SK}, C_{DS}, C_{SD}>$ indexed by unordered pairs $<S, D>$. Note that only one such tuple is stored for a given pair of peers involved into (possibly multiple) sessions routed through the node. Let K be the node storing the tuple. Then h_{DK} is the number of hops between D and K made by the last packet seen by K and traveling from D to S, h_{SK} is the last-seen number of hops on the path from S to K, C_{DS} and C_{SD} are the discard counts calculated by the rule and applicable to the two forwarding directions.

Whenever K sees a packet traveling from S to D, it updates h_{SK} and sets:

$$C_{DS} = m_b \times [(h_{SK} + h_{DK}) - h_b]$$

Similarly, for a packet traveling from D to S, K updates h_{DK} and sets:

$$C_{SD} = m_b \times [(h_{SK} + h_{DK}) - h_b]$$

Note that the subscript b (in m_b and h_b) is interpreted relative to the actual source of the incoming packet and it refers to the opposite direction. In both cases, $h_{SKD} = h_{SK} + h_{DK}$ corresponds to the current length of the path connecting S and D and passing through K, as perceived (expected) by the incoming packet. If the value of h_b in its header is less than h_{SKD}, it means that there exists (or perhaps existed a short while ago) a path shorter than the one passing through K. Thus it is highly likely that the packet will reach its destination faster via another path, and its current copy will be discarded further along its route as a duplicate. Consequently, the node may consider dropping the packet, with the confidence of this decision being proportional to the difference between h_{SKD} and h_b.

Suppose that m_b is 1 and the packet travels from S to D. Then if $C_{SD} > 0$, the route leading through K can be suspected of being superfluous and suboptimal. A careful analysis of the possible scenarios leads us to the observation that it makes sense to follow up on this suspicion and drop the packet only if both C_{SD} and C_{DS} are greater than zero. Otherwise, the neighbors of K, which may also be located on a superfluous path, will not learn about that fact and may keep forwarding other copies of the packet received from elsewhere.

The two values, C_{SD} and C_{DS} are viewed by the rule as counters. Whenever a packet traveling from S to D (the other direction is symmetric) finds both counters positive, the rule decrements C_{SD} and indicates that the node is not eligible (the packet is dropped). Thus, the higher the value of C_{SD}, the more consecutive packets trying to reach D via K will be dropped before forwarding in that direction is (tentatively) resumed by the node. By using a factor m_b, which may be less or greater than 1, the rule may be more aggressive with avoiding suboptimal routes, or more fuzzy, i.e., allow alternative routes and drop fewer packets. Note that regardless of the actual value of m_b, as

long as it is greater than zero, drastically suboptimal routes are discouraged more than those that are only slightly suboptimal. Also, regardless how suboptimal the route appeared at some point, it will be tried again at some later time, with less suboptimal routes being reconsidered more often. Additionally, the frequency of those attempts depends globally on the mobility factor m_b. They are critical from the viewpoint of responding to the changing routing opportunities in a dynamic network.

As implemented in TARP, the SPD rule is augmented by a simple load balancing (LB) mechanism that makes the values of the discard counters depend not only on the suspected deviation of the path length from the optimum, but also on the intermittent congestion level at the node. The rationale behind this approach is that it may be sensible to forward a packet via a longer path, if the shorter path appears to be congested. This way, the rule attempts to balance the load among the paths that are close (albeit not necessarily equal) in terms of the number of hops. With this modification, the formulas for calculating the discard counters take the shape:

$$C_{SD/DS} = (m_b + d \times t_t) \times [(h_{SK} + h_{DK}) - h_b]$$

where d is a constant coefficient dubbed the *diversity factor*, and t_t is the last sample of the packet processing time. By making the overall factor depend on t_t rather than the average packet processing time t_{avg}, the rule is intentionally "jumpy" with interpreting the load fluctuations at intermediate nodes. Depending on the setting of d, this approach tends to balance the routing among close paths and also helps in those situations where there are multiple shortest paths between a given source and destination. Instead of using all those paths at once (which would happen with unmodified SPD), the new rule may alternatively choose only one (or some) of them.

It is worthwhile to mention that the end nodes participating in a session will also maintain a separate buffer called *SPD Buffer* to cache some useful information. In particular, a source node keeps one entry per destination in the SPD buffer. Thus if a source S transmits to three different destinations (e.g. three separate application agents are attached to three different ports of a station), three different entries will be kept. The format of a particular entry is : $<D, h_b, m_b>$, where, D is the destination identifier, h_b is the number of hops traveled by the latest packet from D to S and m_b is the mobility factor for sessions from S to D.

3.2.5 Data link layer issues

After receiving a packet from the Network Layer, the task of the Data Link Layer is to broadcast the packet into the medium. All neighbors within the transmission range will receive the packet. If the Data Link Layer has the provision of sending acknowledgments, then all receiving nodes will send an acknowledgment as soon as they receive the data packet. Upon receiving an acknowledgment, the sender's Data Link Layer should immediately notify the Network Layer about the success, e.g., by sending the *signature* of the packet for which it has received the acknowledgment.[5] On the

[5]This calls for cross layer interaction.

other hand, if the sender's Data Link Layer receives no acknowledgment for a predetermined time period, then it will re-broadcast the data packet. If the retransmission of packet is unsuccessful for a predetermined number of attempts, the Data Link Layer will notify the Network Layer about the failure. This may happen if the source effectively has no neighbors at all.

If the Data Link Layer has no provision for acknowledgments for broadcast packets[6], then the Network Layer should take the responsibility for sending acknowledgments. Many variations for sending acknowledgments may be built on top of the Data Link Layer. One option is to force the Network Layer to explicitly send a unicast acknowledgment packet to the sender with a copy of the packet *signature*. This will generate several acknowledgments for a single broadcast.

Another simple but unreliable scheme is to use *passive acknowledgments* in place of true acknowledgment. To determine whether a packet was successfully forwarded, the network layer of the sender would listen for the first copy of the packet retransmitted by a neighbor. This simple and (almost) free acknowledgment technique works for all hops except the last, because a packet is not forwarded beyond its destination. We can alleviate this problem by forcing the destination to send back an explicit acknowledgment or re-broadcast the packet one more time.

In Section 3.3 we will show how a certain type of acknowledgment can be incorporated into the Data Link Layer.

3.2.6 Cache management

In this subsection we will briefly describe how the entries in *DD Cache*, *SPD cache* and *SPD Buffer* are added, updated and deleted. Upon receiving a packet, if the packet *signature* is already present in the *DD Cache*, then the entry is refreshed (i.e., the expiry time is updated), otherwise a new entry with this *signature* is added. If the *DD Cache* is full, then an existing entry should be deleted. The first choice for deletion will be the oldest entry of the same session to which the new packet belongs. If the *signature* is of the first packet of a session, then the globally oldest entry is deleted.

As soon as a particular entry in the *DD cache* expires, it is removed. If this is the last entry concerning the session from a source S to a destination D then the corresponding entry from the *SPD buffer* must also be deleted at source S. There is no point in keeping any information concerning a session which most likely has already ended.

However, deleting an entry with source S and destination D in *SPD cache* requires that there is no entry with S and D or D and S as the source and destination of any entry in *DD cache*. An entry in *SPD cache* keeps information related to packets flowing from S to D and from D to S. As long as at least one entry for this pair of stations remains in the *DD cache*, indicating that a session between the two stations may be still active, the corresponding entry can not be deleted from the *SPD cache*.

[6]IEEE 802.11 is in this category.

3.2.7 Protocol parameters and their initial values

TARP has only six parameters whose initial values can be easily set. The first two are maintained by the protocol itself. The protocol tries to keep their values optimal. The last four might be included in protocol's modes of operation and be controlled by the application designer or, dynamically, by the application itself.

1. **Average transmission time t_{avg}:** Assuming knowledge of the maximum packet length L and network's capacity C, a rough estimate of transmission time for a single packet is $\frac{L}{C}$. Hence the average transmission time t_{avg} may be approximated by following equation:

$$t_{avg} = c \times \frac{L}{C}$$

   where, $c > 1$. The protocol will adjust t_{avg} to move the value closer to the actual average time. The value of t_{avg} is also used for calculating the expiry time of *DD cache* entries. Setting $t_{avg} = 0$ will not cause a protocol's failure. The only outcome of initializing it to zero is the premature expiry of the first few entries in the *DD cache* which leads to more copies of the packets these entries correspond to circulating in the network.

2. **Transmission time buffer expiry time T_b:** Assuming that the initial value of t_{avg} is positive, $T_b = c \times t_{avg}$, where $c > 1$. Note that $T_b >> 0$, otherwise the protocol would never be able to calculate a good estimate of t_{avg}.

3. **Parameter C_a used in calculation of the average transmission time t_{avg}:** The value of C_a emphasizes either the past or recent measurements of t_{avg}. A station will react more slowly to changes in the actual transmission time if C_a is large. Depending on the environment in which the protocol operates, different values may become optimal. Initial good values range between $0.5 - 0.8$. An initial setting of 0.65 is acceptable.

4. **Flooding control constant F_c:** The capability of discarding duplicate packets is determined by the value of F_c. A higher value of F_c (close to 1) will require high memory but will provide higher duplicate discard capability. On the other hand, lower value of F_c will require less memory but will lower duplicate discard capability. A value of $F_c = 0.75$ seems to be a good compromise between memory requirement and duplicate discard capability.

5. **Mobility factor $m_{f/b}$:** The mobility factor $m_{f/b}$ can be set to 1 initially, because the discard counters (C_{SD} or C_{DS}) calculated with this value closely approximate the difference (in hops) between the current path and the shortest path.

6. **Diversity factor d:** The diversity factor determines how diversified paths are chosen by the *LB* algorithm. A lower value indicates less diversification hence shorter paths are preferred; higher values indicate the opposite. Empirical evidence shows that a value of 0.3 is optimal and can be chosen initially.

3.2.8 Mode of operation

The protocol may operate in various modes and its mode of operation can easily be changed. An argument of the protocol would determine the complexity desired by the application.

For example, let us see how we can control flooding at different levels of complexity. In the simplest case, the flooding can be controlled only by the *time to live* field r. This can be achieved by setting $F_c = 0, m_b = 0$ and $d = 0$. Every station would simply forward packets addressed to other stations, and would receive and remove those packets that circulate in the network too long (based on r and h fields provided by the higher levels). A second level of control is accomplished by setting $F_c > 0, m_b = 0$ and $d = 0$. This will enable the duplicate discard algorithm but disable SPD and LB. In other words, this will control the number of duplicates traveling in the network but will not limit the number of packets on sub-optimal paths and will not affect the path diversity. The third level of control can be achieved by setting $F_c > 0, m_b > 0$ and $d = 0$. This will control the number of duplicates wandering in the network and taking sub-optimal paths but does not control path congestion. The fourth and final level of control can be achieved by setting $F_c > 0, m_b > 0$ and $d > 0$, which will turn on the full control.

The required complexity level of TARP can be determined by several factors, including the amount of resources at a node (memory, computing power), the degree of stability of paths (the impact of mobility and general dynamics of nodes), the expected reliability and bandwidth of the network (quality of service). Different parameterized combination of the above four modes allows the designer of an ad hoc application to tailor TARP to the needs of the application and its environment.

3.2.9 Simulation results

We used a detailed simulation model based on *ns*-2 [1] to evaluate the performance of our protocol. The wireless extension of *ns*-2 supports the simulation of multi-hop wireless networks complete with physical, data link and MAC layer models [6]. The distributed coordination function (DCF) of the IEEE standard 802.11 [13], was used as the MAC layer. *Broadcast* data packets are sent using physical carrier sensing and are not acknowledged. The radio model uses characteristics similar to a commercial radio interface resembling Lucent's WaveLAN [14, 62]. WaveLAN is a shared-media radio with a nominal bit rate of 2 Mbps and a nominal radio range of $250m$.

The protocol's performance was not measured on a particular workload taken from real life, but rather under a range of conditions. Our initial protocol evaluations are based on the simulation of 25 wireless nodes forming an ad hoc network, and moving over a square area ($670m \times 670m$) of flat space for 500 seconds of simulated time. Each run of simulation accepts a *scenario file* that describes the exact motion of each node (a *movement file*) and the exact sequence of packets originated by each source (a *traffic generator file*). We pre-generated 77 different *scenarios* through various combinations of 11 different *movement patterns* with 7 different *movement files* for each

pattern.

There was no buffering of packets by the TARP *routing agent*. All packets sent by the routing layer were queued at the *interface queue* until the MAC layer could transmit them. The interface queue was a priority queue with a maximum size of 50 packets.

3.2.9.1 Mobility model

The mobility model uses the *random way point* model [6] in a square region of size $670m \times 670m$. The movement scenario files are characterized by a *pause time*. Each node begins the simulation by remaining stationary for *pause time* seconds. It then selects a random destination in the $670m \times 670m$ flat space and moves to that destination at a speed distributed uniformly between 0 and $10m/sec$. Upon reaching its destination, the node pauses again for *pause time* seconds, selects another destination, and proceeds there as previously described, repeating this behavior for the duration of the simulation. Each simulation ran for 500 seconds of simulated time.

We ran our simulations with movement patterns generated for 11 different pause times: 0, 50, 100, 150, 200, 250, 300, 350, 400, 450, 500 seconds. A pause time of 0 seconds means continuous motion and a pause time of 500 seconds means no motion of any node.

3.2.9.2 Traffic model

Traffic sources were CBR (constant bit-rate). The source-destination pairs were spread randomly over the network. The traffic generation was bidirectional. For any particular pair (S, D) there were at least two active sessions. One session was flowing traffic from S to D and the other was a reverse session from D to S. Only 512-byte data packets were used. The number of source-destination pairs and the packet generation rate in each pair was varied to change the offered traffic load in the network.

We experimented with different number of sessions and different packet generation rates. The connections were started at times uniformly distributed between 0 and 180 seconds.

3.2.9.3 Performance metrics

The following two performance measures were taken:

(i) Packet delivery fraction (PDF): This is defined as the ratio of the number of data packets successfully delivered to the destination to those generated by the CBR sources.

(ii) Average end-to-end delay: This is defined as the ratio of total delay of all data packets from the moment of their generation to the moment of delivery, to the total number of those packets received by their destinations.

The packet delivery ratio is important in the sense that it provides a picture of the perceived loss rate of packets by the transport layer. Average delay will depict the response time of the network

perceived by the transport layer. Note that, these two performance metrics are inter-related and each of them has impact upon the other. For example, lower packet delivery ratio means less packets were delivered to the ultimate destinations. For average delay measurement, only those packets that have reached the destinations are counted. Dropped packets are not considered for delay measurement. Probabilistically speaking, the longer-delayed packets (traveling over longer paths) are more likely to be dropped. On the other hand, the packets traveling shorter paths are highly probable to reach their destinations. Short path length packets suffer less delay, lowering the average delay. Higher routing loads cause higher congestion, and increase the number of packet drops at interface queues due to the limited queue size. This causes a lower packet delivery ratio and a higher average delay.

3.2.9.4 Simulation setup at a glance

Table 3.1 summarizes all the simulation configurations that were discussed in the previous sub-sections.

SIMULATION CONFIGURATION

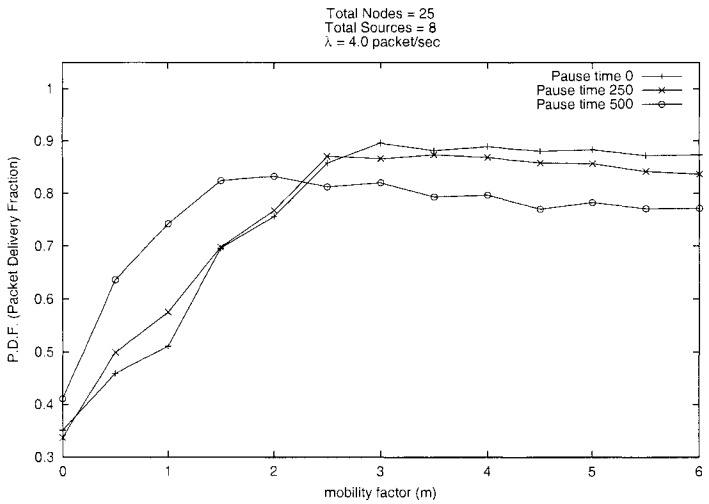
Application Layer	CBR
Routing Protocols used	TARP, DSDV, DSR, AODV
MAC Layer	IEEE 802.11
Radio Propagation Model	Free Space and Two-ray Ground Reflection
Network Interface	Lucent WaveLan
Traffic Pattern	Bidirectional CBR streams
Packet Size	512 bytes
Bandwidth	2 Mbps
Transmission Range	250m
Node distribution	Uniform
Mobility Model	Random Waypoint
Velocity	0-10 m/s
Deployment Region	Square ($670m \times 670m$)
Simulation time	$500s$

Table 3.1: Configuration used in simulation

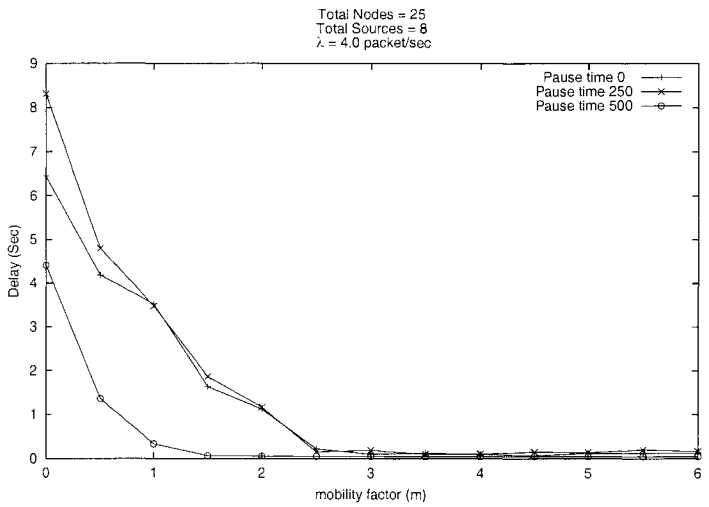
3.2.10 Effect of the mobility factor

Our first set of experiments was designed to check the impact of the mobility factor on performance. For a particular *scenario* we first varied the mobility factor (m_b). We tried to figure out a "good" value of the mobility factor which was global (i.e. same for every node) and for which the over all network achieved peak performance. For every scenario, we found the value of the mobility factor which achieved this goal.

Figure 3.1 illustrates the most relevant properties of TARP at a glance by showing the impact of the mobility factor m_b on two performance measures: the *packet delivery fraction* (PDF) and the *end-to-end delay* averaged over all received packets. Three mobility scenarios were considered:



(a)

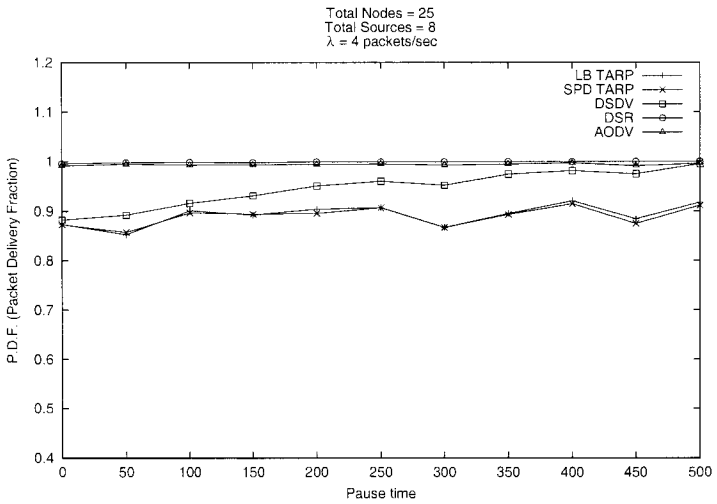


(b)

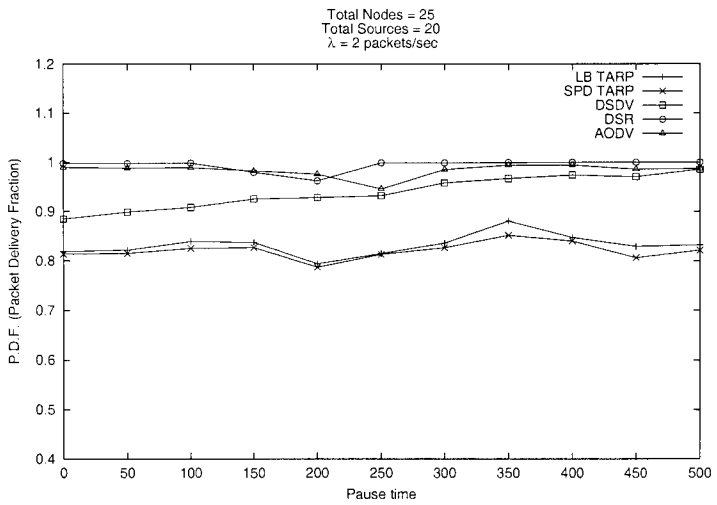
Figure 3.1: (a) Effect of mobility factor on packet delivery ratio. (b) Effect of mobility factor on average delay.

continuous movement (pause time = 0), average mobility (pause time = 250), and the stationary case (pause time = 500).

Note that the "best" value of m_b need not be hardwired into the protocol or even into the application. A source can initially set $m_{f/b}$ to a small value (e.g., 0) and increase it gradually while monitoring the quality of service received by the session. Depending on the dynamic configuration of the network, different values of m_b may provide the best performance. Figure 3.1 indicates that the best value of m_b typically lies between 1 and 3.



(a)



(b)

Figure 3.2: Comparison of TARP in terms of PDF with other protocols (a) 8 sources (b) 20 sources

3.2.11 Comparison with other protocols

Figure 3.2 shows the PDF in TARP, DSDV, AODV and DSR for two different numbers of sources (sessions). Two versions of TARP are shown: the version labeled as SPD operates without the LB rule, while the LB version augments the SPD rule by the load balancing extension. Thus, the four graphs also illustrate the impact of the LB extension on the PDF in TARP. While TARP is slightly worse in these metrics than the other protocols, its performance is at least comparable to that of the established- and considerably more complex-solutions.

3.2.11.1 Comment on performance

In Figure 3.2, the gap separating TARP from the other protocols becomes narrower when a small number of nodes is contributing to network traffic and tends to widen for a larger number of sources. Essentially, there are two reasons why TARP yields to the competition. First, the path convergence of TARP to the single "best" path between source and destination is not perfect, regardless of the setting of the mobility factor $m_{f/b}$. While TARP is good at identifying shortest paths, it does not cope well with multiple shortest paths, if they happen to be present. The LB addition to the SPD rule exhibits a tendency to switch among multiple shortest paths instead of using them simultaneously, but this isn't perfect. The second problem is the inadequacy of the collision avoidance scheme in the MAC layer. As all communication is inherently broadcast, a forwarding node cannot take advantage of data-link acknowledgments to improve the reliability of communication. We will address this issue in the next section.

3.3 Proposed improvement of the MAC layer

A forwarding node in TARP would like to know whether the packet has been picked up by any eligible node in the neighborhood. It is also okay if several nodes considering themselves eligible pick up the same packet. The identity of those nodes is of no direct importance to the forwarding node. However, with the traditional collision avoidance scheme of IEEE 802.11, the forwarding node has no means of determining or even guessing at the success of its broadcast transmission in the data-link layer. The approach used in our first implementation of TARP was to listen for a copy of the transmitted packet (forwarded by an eligible node) and use it as an indication of success—in addition to timers used to diagnose failures. There are two problems with this solution. First, depending on the load at the eligible node, there can be a significant delay between packet reception and retransmission. Second, to make this idea work, the destination itself has to "forward" (i.e., retransmit) all received packets, which creates unnecessary noise in its neighborhood.

In this section we present an innovative idea of "fuzzy acknowledgments" and with the aid of it, we propose a novel mechanism for "retransmission" of broadcast packets by IEEE 802.11 MAC. We also show how this idea can significantly improve the protocol's performance.

3.3.1 Deficiencies of IEEE 802.11

The IEEE 802.11 family of MAC schemes is reasonably well equipped to handle point-to-point communication in the face of multiple parties trying to talk at about the same time [13]. With the four-way handshake, RTS/CTS/DATA/ACK, the sending node first verifies that the other party is present and willing to receive, and reserves bandwidth for the actual data exchange, and then, following the packet transmission, it receives an acknowledgment from the recipient. The RTS/CTS part of the handshake also accounts for the "hidden terminal" problem by including the recipient in

the bandwidth reservation part of the complete exchange.

Unfortunately, none of these features is available for broadcast transmission, particularly in its flavor needed in TARP. First, the RTS/CTS part makes no sense because 1) the recipient's identity is unknown and unimportant, 2) there can be multiple legitimate recipients that do not know about each other. Second, even the two-way handshake, DATA/ACK, is not possible because of 2. Consequently, the only available option is to transmit blindly, following the standard DIFS delay and back-off procedure prescribed by the scheme. This completely ignores the hidden terminal problem, greatly increases the likelihood of a collision, and renders the data exchange highly unreliable. Note that one naive approach to rendering multi-point communications more reliable is to unicast a separate copy of the packet to each neighbor. The inefficiency of such a technique has been demonstrated in [9].

3.3.2 The proposed mechanism

We propose the following simple solution as an extension of the IEEE 802.11 MAC protocol. When a node receives a packet for which it considers itself eligible, it waits for a short amount of time, defined by the short inter-frame space (SIFS), and then sends an acknowledgment. When multiple recipients send their acknowledgments at (almost) the same time, the sender will not be able to recognize them as valid packets. However, the sender can interpret any activity (of a certain bounded duration) that follows the end if its last transmitted packet as an indication that the packet has been successfully forwarded. Although the value of this indication is inferior to that of a "true" acknowledgment, it may provide the kind of feedback needed by the data-link layer to assume that its responsibility for handling the packet has been fulfilled.

Thus, having completed a packet transmission, the sender immediately switches to listening mode and awaits a period of silence (of duration comparable to SIFS) followed by a burst of activity (of duration comparable to the duration of an acknowledgment packet). If such an event occurs within the prescribed interval, the sender assumes that the packet has been passed over; otherwise, it schedules a retransmission. With this approach, the acknowledgment packet (which carries no information other than its presence) can be made very short and consist of some characteristic pattern unlikely to be encountered in a regular packet.

3.3.3 Incorporation into TARP: the third rule

As the role of a fuzzy acknowledgment in TARP is to tell the sender that its packet has been forwarded towards the destination, it is important that only those recipients that are actually going to forward the packet (or the destination itself) send the acknowledgments. Consequently, acknowledgments cannot be sent mechanically in the data-link layer, and the incorporation of fuzzy acknowledgments into TARP requires some cross-layer coordination. For illustration, consider the configuration of nodes shown in Figure 3.3. Assume for simplicity that the network is static and that there is a

session in progress between nodes 1 and 2, with the current converged path passing through nodes $<1, 10, 15, 20, 22, 2>$. When node 1 sends a packet in the first hop, it might be received by nodes 6,7,10,12, and 13. In the next hop, node 10 is going to rebroadcast the packet because it lies on the converged path. But what will happen if the packet is received by some neighbors of node 1 but not by node 10, e.g., because of an interference. If any of those neighbors sends an acknowledgment that is subsequently received by node 1, then node 1 will conclude that the packet has been forwarded, which, of course, is not the case.

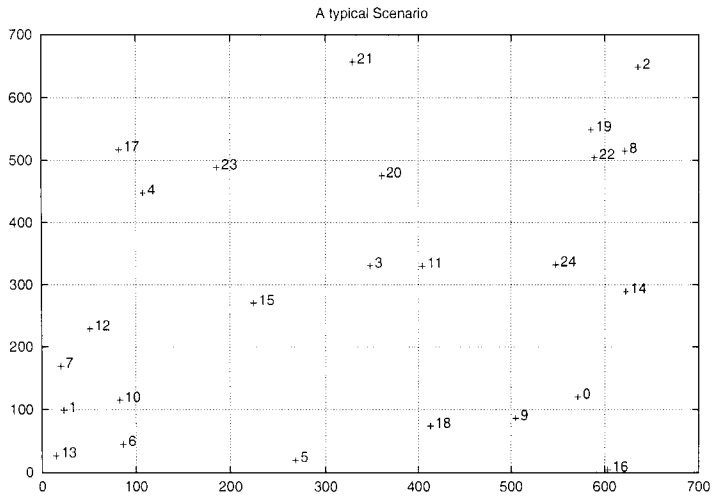


Figure 3.3: Showing a random node configuration

To see another problem, suppose that a packet sent from node 1 to node 2 has arrived at node 4, which, according to its current perception of path convergence, considers itself eligible. The recipients of the transmission of node 4 are nodes 17 and 23. If node 17 decides to forward this packet again, it will arrive back at 4 and 23, where it will be recognized as a duplicate. Thus, neither of the two nodes will find itself eligible and they will send no acknowledgments. Consequently, node 17 will keep retransmitting the packet over and over until it finally decides that the optimal path for the session lies elsewhere. This will create unnecessary activity in the neighborhood of node 17 contributing to the overall interference and reducing the amount of usable bandwidth.

By blurring the distinction between layers a bit further, we can avoid this problem and turn it into one more heuristic facilitating path convergence in TARP. Consider three nodes: the source S, the destination D, and an intermediate node N. Suppose that Δ refers to a time interval and define:

$$RF_{SND}(\Delta) = \frac{n^{ack}_{SND}(\Delta)}{n^{fwd}_{SND}(\Delta)}$$

where, n^{fwd}_{SND} is the total number of packets between S and D passed through N within time Δ, and $n^{ack}_{SND} \leq n^{fwd}_{SND}$ represents the portion of those packets for which N has received (fuzzy) acknowledgments. Depending on the setting of the interval Δ, RF (called the *relevance factor*) can

be viewed as a measure of N's relevance in forwarding packets between S and D. Alternatively, we can view RF as a measure of probability that N lies on the optimal path between the two end-nodes, at least as long as the configuration of nodes remains static.

With the mobility included, the indications of RF become less accurate. This is not solely a problem of TARP: any information related to the configuration of paths tends to become outdated, if the nodes are allowed to change their location. Thus, the proper way to interpret the value of RF should be determined experimentally. One natural idea is to use a threshold. A value of RF above the threshold indicates that the node is relevant in sustaining the session.

A straightforward way to implement the new rule is to take advantage of the DD cache and flag those packets for which acknowledgments have been received with one extra bit. This approach automatically equates Δ with the DD cache expiration interval and rids the protocol of one parameter.

3.3.4 The improvement

Figure 3.4 illustrates how the relevance factor RF affects the performance of TARP in terms of the packet delivery fraction. It shows that RF does influence the quality of routing and hints at the range between 0.6 and 0.75 as the suggested setting. Notably, the same range of values seems to be adequate for different mobility levels, which allows us to make RF a constant rather than a dynamically tunable parameter.

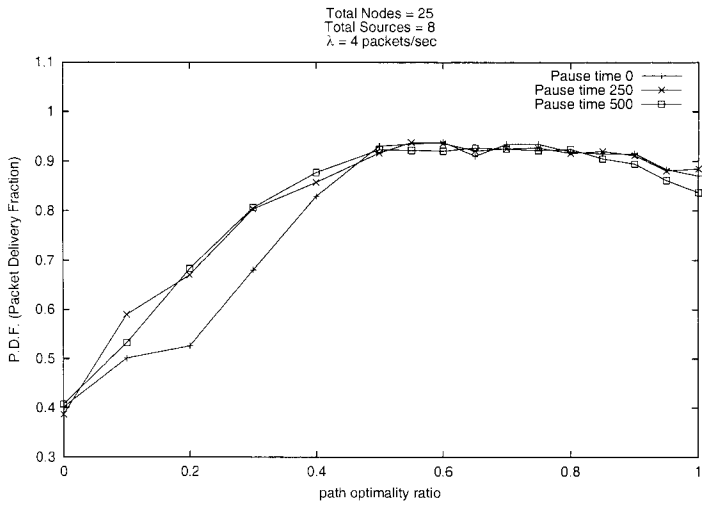
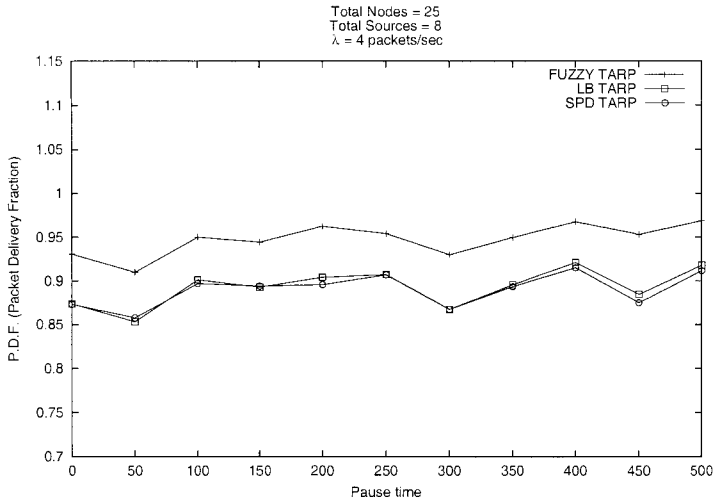


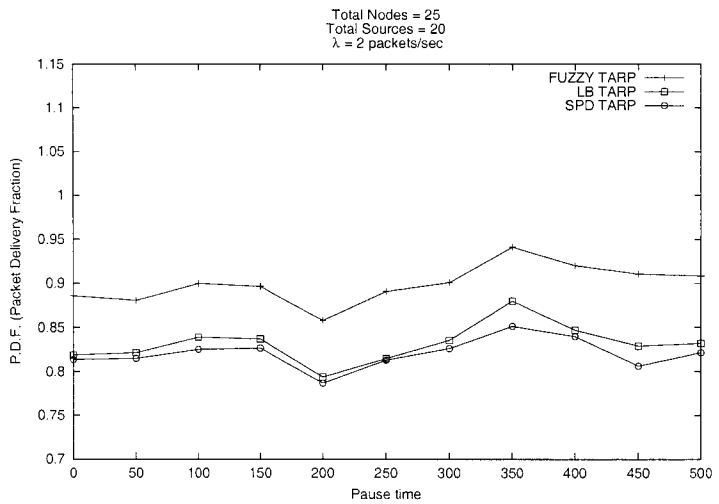
Figure 3.4: Impact of threshold on packet delivery fraction

In Figure 3.5, we show how much improvement has been brought into TARP by the addition of the new rule in terms of packet delivery ratio. Figure 3.5 (a) presents a magnification of the two TARP curves from Figure 3.3 (a) with the inclusion of a new curve reflecting the fuzzy-acknowledgment version with $RF = 0.7$. The magnitude of the observed improvement has been consistent across different mobility and traffic levels. The bottom portion of Figure 3.5 compares

the three variants of TARP for a larger number of sessions.



(a)

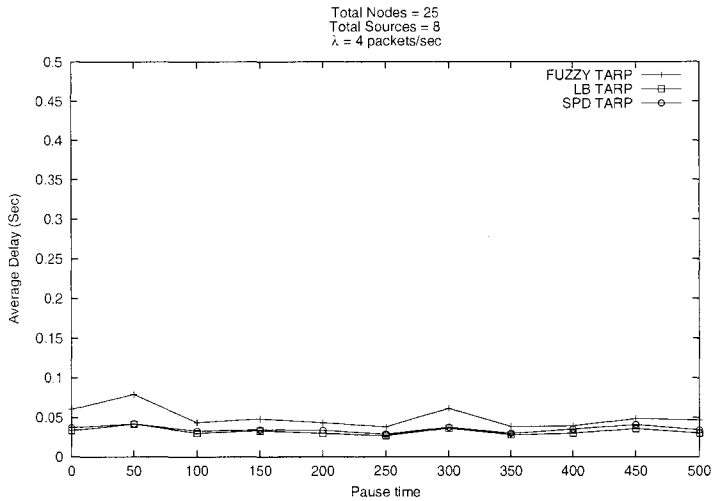


(b)

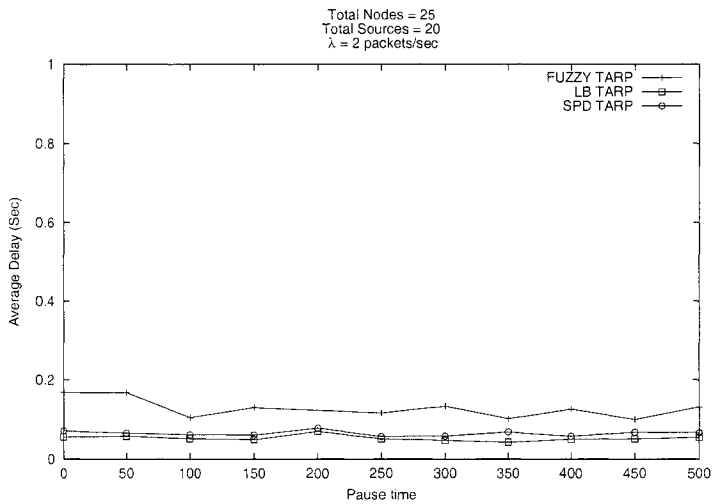
Figure 3.5: The performance improvement due to the addition of fuzzy acknowledgments (a) 8 sources (b) 20 sources

Figure 3.6 shows the comparison curve in terms of average delay in the new version as compared with the other two versions. Again two graphs are plotted with different traffic loads. Figure 3.6 (a) shows 8 sessions and Figure 3.6 (b) shows 20 sessions. The average delay is slightly higher after the inclusion of fuzzy acknowledgment. Recall that the fuzzy acknowledgment mechanism has provided the retransmission capability of TARP at the MAC level which does not exist in the other two versions of the protocol. Although the average delay is higher in the new version of the protocol, the overall delay will be lower. The retransmission mechanism in the new version will allow TARP to deliver more packets without any intervention of the upper layers. Those undelivered packets in

the other two versions of the protocol will be ultimately retransmitted by the upper layers with even higher delays, which is not reflected in the curves.



(a)



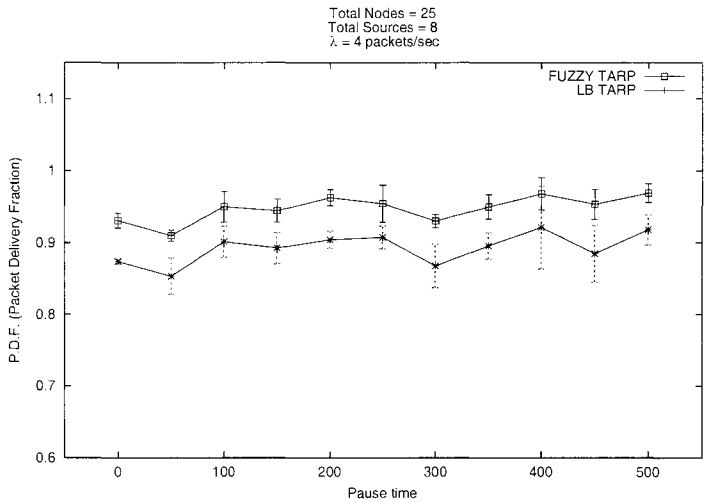
(b)

Figure 3.6: Comparison of average delay with (a) 8 sources (b) 20 sources

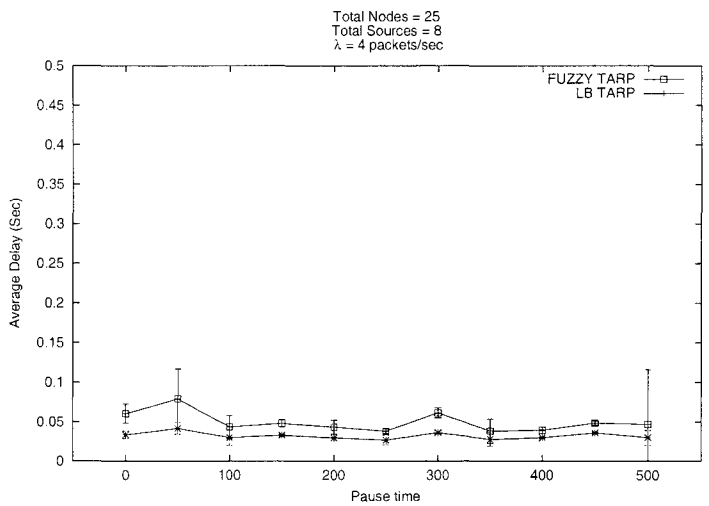
3.3.5 Confidence intervals

In our experiments, we selected 11 different *movement patterns* with the *pause time* of: 0, 50, 100, 150, 200, 250, 300, 350, 400, 450, and 500 seconds. Here a pause time of 0 meant continuous motion of all nodes and a pause time of 500 meant no movement of any nodes. For each movement pattern, we generated 7 different scenario files and took the average value of the performance metrics. In this section we show the variability of the measurements by giving the confidence intervals for the data. Figures 3.7 and 3.8 show the 95% confidence intervals of two performance metrics, PDF

and Average Delay, with 8 and 20 sessions. Two versions of the protocol are plotted, one with load balancing (LB TARP) and the other with the addition of fuzzy acknowledgments (FUZZY TARP).



(a)

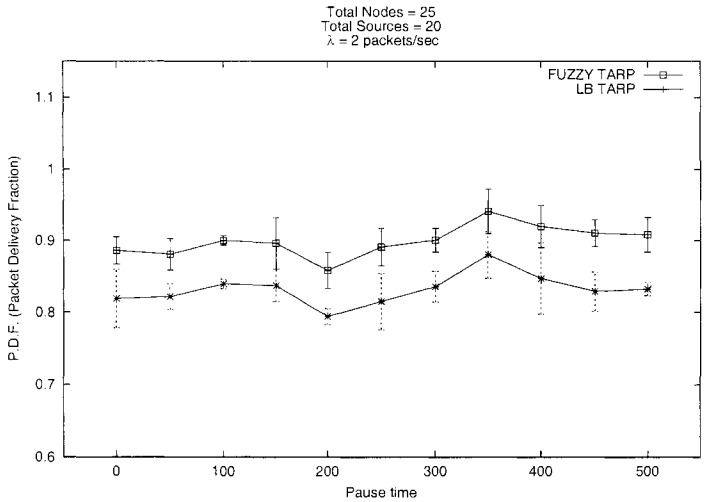


(b)

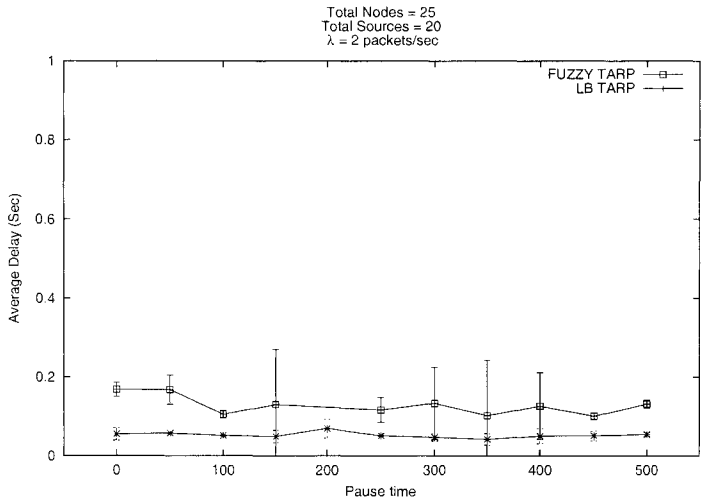
Figure 3.7: Confidence intervals of (a) packet delivery ratio (b) delay with 8 sources on two version of the protocol

Figure 3.7(a) and Figure 3.8(a) show that PDF has relatively small confidence intervals for both versions of the protocol. We can also conclude that the PDF of the protocol with fuzzy acknowledgments is always better than that of the protocol without fuzzy acknowledgments.

On the other hand, Figure 3.7(b) and Figure 3.8(b) show that the confidence intervals of average delay are larger with fuzzy acknowledgments than without.



(a)



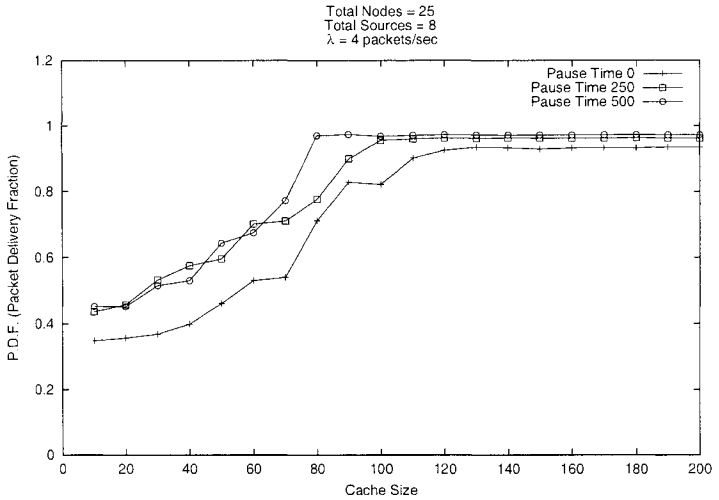
(b)

Figure 3.8: Confidence intervals of (a) packet delivery ratio (b) delay with 8 sources on two version of the protocol

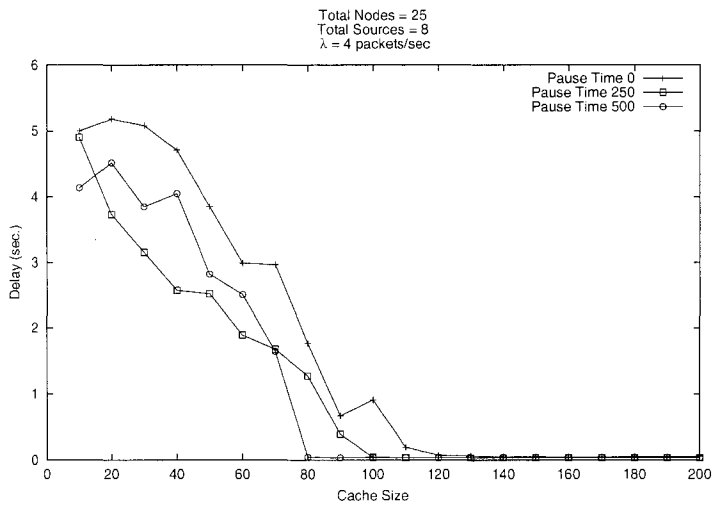
3.3.6 Effect of cache size

The DD cache is used by the first rule described in Section 3.2.3. Nodes running our protocol do not create any routing tables—no information about the path is kept. Instead, every node has a *DD cache* whose purpose is to keep certain entries that identify the passing packets. The entries of *DD cache* are the *signatures* of the previously passed packets. Keeping such signatures in the *DD cache* allows a node to detect duplicate packets (possibly) coming in the future and avoid re-forwarding those duplicates.

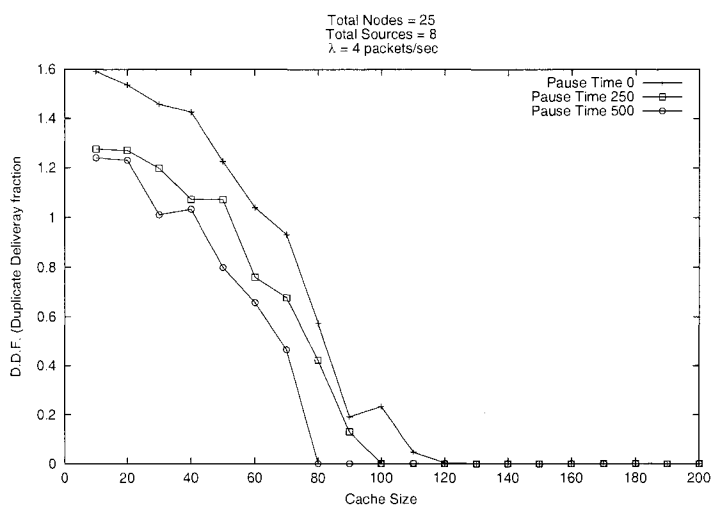
The size of the *DD cache* has a significant impact on the performance of the protocol. Making the size too small will cause some packets to remain undetected as duplicates and may waste bandwidth



(a)

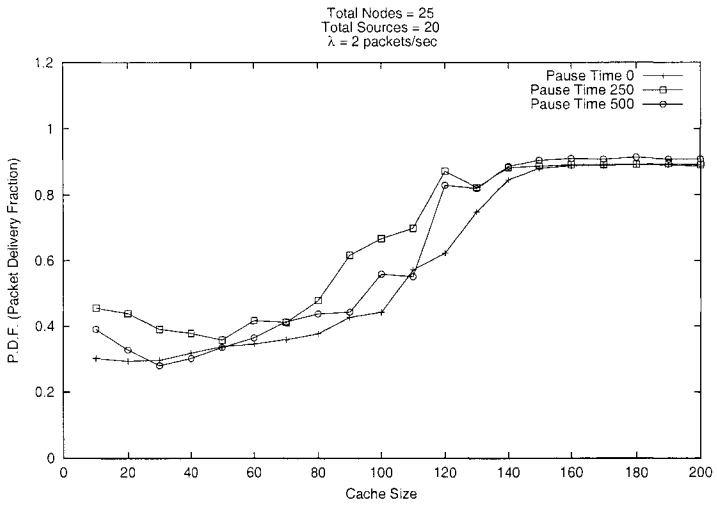


(b)

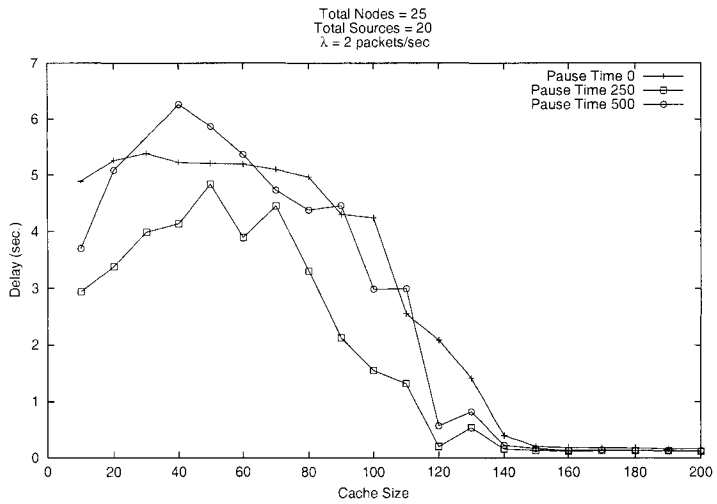


(c)

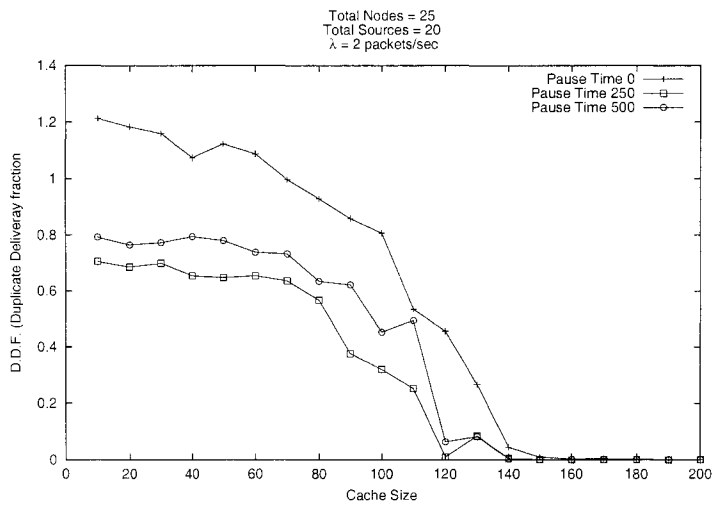
Figure 3.9: Effect of cache size on (a) p.d.f. (b) average delay and (c) d.d.f.



(a)



(b)



(c)

Figure 3.10: Effect of Cache size with increased load

by causing the nodes to re-forward the same packets. In this section we will briefly present the experimental results showing the impact of the *DD cache size* on network performance.

In addition to the two performance metrics PDF and average end-to-end delay, we define a third metric called *DDF (Duplicate Delivery Fraction)* as follows:

Duplicate Delivery Fraction (DDF) This is defined as the ratio of the number of duplicate data packets delivered to the destination to the number of packets generated by the sources.

Note that DDF is a unit-less quantity and a value of '0' indicates that the nodes were successfully able to detect all duplicate packets and no destination node received any packet more than once.

Figure 3.9 shows the effect of cache size with 8 sources on the three performance metrics: PDF, Average delay and DDF. All three graphs were plotted with three mobility patterns: continuous mobility (with pause time 0), average mobility (with pause time 250) and no mobility (with pause time 500). For each mobility pattern 5 scenario files were pre-generated and the average performance metrics of all scenarios were measured.

Figure 3.9(a) shows the effect of cache size on packet delivery fraction (PDF). With a smaller cache, fewer packets were successfully delivered. With an increase in the cache size, the number of successfully delivered packets also increased significantly. A node in an ad-hoc wireless network may take on one of three different roles—(i) it may act as an originator of packets, (ii) it may behave simply as an intermediate forwarding node, or (iii) it may act as both. Therefore, depending on its role, a node may inject two kinds of traffic into the network: 1) its own packets to be delivered to some destination (if any) and 2) some other node's packet to be delivered to some other destination node(s). The number of packets of the second kind depends on each node's duplicate detection capability. The duplicate discard rule does not deliver a packet to the data link layer for a possible transmission if it has seen the packet before. With a smaller cache, fewer packet signatures of previously passed packets can be stored, reducing the capability of intermediate nodes to detect duplicate packets. As a result, a packet may be supplied to the data link layer more than once. In the mean time, the node continues to receive more new packets for forwarding. All the new packets and duplicate packets are stored in the same FIFO queue by the data link layer. Some new packets are dropped due to the fact that queue size is limited at the data link layer. Further reduction of the cache size will cause the data link layer to spend much of its effort in forwarding duplicates rather than new packets.

Figure 3.9(b) shows the effect of cache size on average end-to-end delay. With a smaller cache, the average delay is higher than with a larger cache size. Again, a significant portion of time is used in forwarding duplicate packets with a smaller cache size causing new packets to be buffered and forced to wait for transmission in the data link layer. This fact causes new packets to be delivered with higher delays.

Figure 3.9(c) shows the effect of cache size on duplicate delivery fraction(DDF). With a smaller cache more duplicate packets were delivered at the destinations than with a larger cache.

Note that, for all three metrics in all three scenarios, the peak performance is achieved when *DD cache* had a storage capability between 80 and 110 packet signatures.

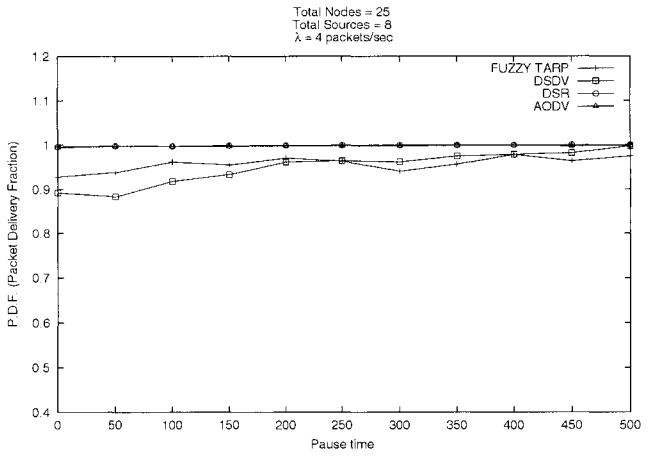
Figure 3.10 shows the effect of the cache size on sessions with 20 sources. This time the peak network performance is achieved with slightly larger cache. All three graphs suggest that the *DD cache size* larger than 120 packet signatures was good enough to achieve peak performance with this type of node population.

In general, smaller cache may lead to suboptimal performance but does not affect correctness of the protocol. A packet is passed through a chain of rules. Depending on the available information, a rule may fail or succeed. If a rule succeeds the packet is dropped, otherwise the packet is passed to the next rule. The lack of information due to limited cache size results in failure of the first rule (DD rule) rather than success. Consequently, some packets would be unnecessarily rebroadcast rather than dropped. This phenomenon will lead to suboptimal performance of the protocol, but the protocol will still be correct from the viewpoint of network connectivity. Therefore, the same protocol can be deployed in nodes with drastically different footprints.

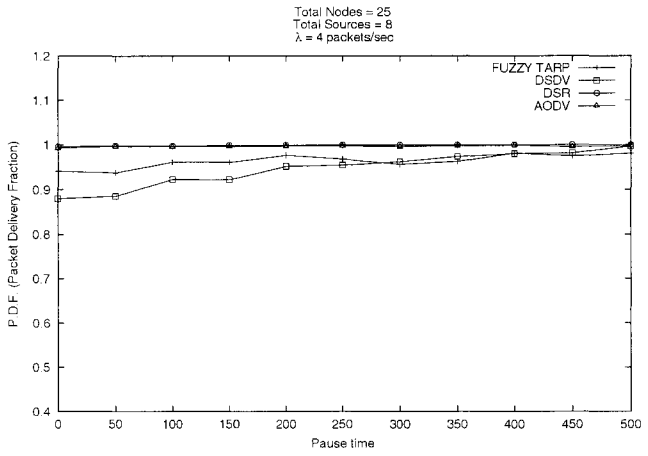
3.3.7 Comparison of the final version of the protocol with other protocols

This section describes the performance of the final version of the protocol to other well-established unicast-based protocols. The performance was evaluated with the most important metric: PDF. We offered varying workload in two different ways—(i) by varying number of packet originators, and (ii) by varying packet sizes. Figure 3.11 (a), (b) and (c) shows the comparison for 8 sources and the packet size of 128, 256 and 512 bytes. Note that the performance of the final version was very competitive with respect to the other (unicast-based) protocols such as DSDV, AODV and DSR. In particular, TARP showed better performance than DSDV in a highly mobile environment (when the pause time < 250) for all different packet sizes. DSDV is slow to react in mobile environments. It periodically exchanges the routing table among the neighbors. Therefore, when a change occurs in a path somewhere in the network, it takes a long time to propagate that information throughout the entire network. During this elapsed time period, the nodes continue to route the packet with previously stored and inaccurate information. Packets are lost on the way due to the use of this inaccurate path information. The performance gap between TARP and other two reactive protocols DSR and AODV was very small (less than 7%).

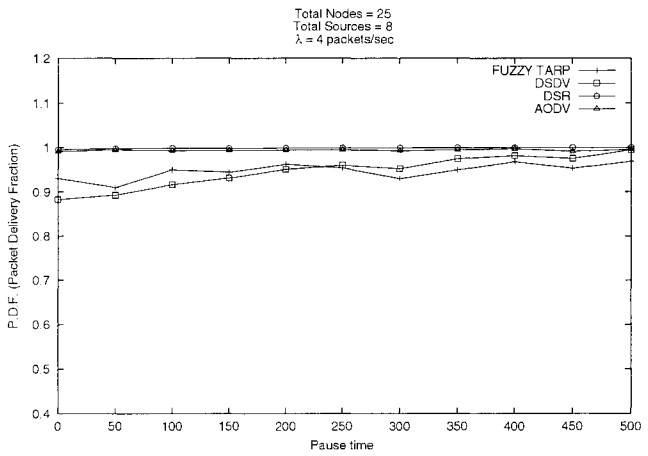
Figure 3.12 shows the performance curve when we increased the workload by increasing the number of packet-generating sources from 8 to 20. Again, three graphs (a), (b) and (c) are plotted with packet size of 128, 256 and 512 bytes respectively. TARP's performance degraded slightly with the increase in workload. Note that with the proposed modification of IEEE 802.11, the reliability of broadcast packets was improved but reliability was still not 100%. The extended reliability feature of unicast packets by the reservation mechanism through short RTS/CTS packet interchanges is still missing for broadcast packets. As a result, broadcast packets are transported with less reliability than



(a)

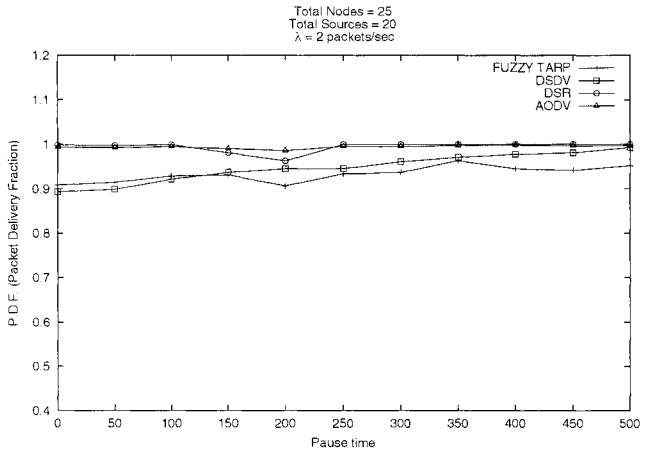


(b)

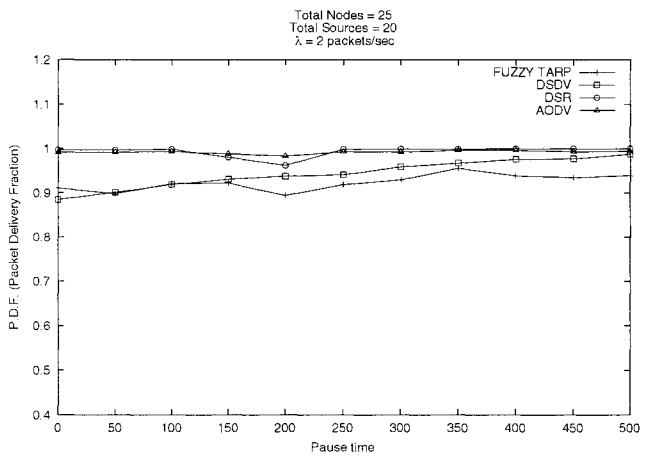


(c)

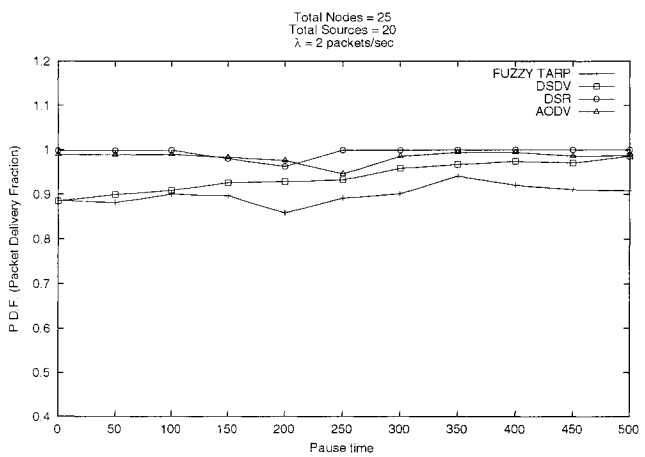
Figure 3.11: Comparison of TARP in terms of PDF with other protocols for 8 sources and packet size of (a) 128 Bytes (b) 256 Bytes (c) 512 Bytes



(a)



(b)



(c)

Figure 3.12: Comparison of TARP in terms of PDF with other protocols for 20 sources and packet size of (a) 128 Bytes (b) 256 Bytes (c) 512 Bytes

unicast packets and may be dropped in the middle of the path on the way to their destination due to collisions. If we could eliminate the hidden node problem for broadcast packets, then perhaps we could improve performance further. So there is still a scope for more work and further performance improvement.

3.4 Chapter summary

In this chapter, we have presented a broadcast-based protocol named TARP. The simple routing protocol discussed in this chapter appeals to us as a promising avenue for deploying maintenance-free ad-hoc networks based on inexpensive and small hardware. Despite its simplicity, TARP, in terms of its performance, can be compared to serious routing protocols with complex route discovery/maintenance mechanisms. With the addition of fuzzy acknowledgments that compensate for the poor handling of broadcast transmissions by the IEEE 802.11 MAC scheme, the gap separating TARP from AODV, DSDV and DSR does not look insurmountable at all, especially when several other enhancements are in store.

Although at the present stage of its development, TARP may appear slightly inferior to protocols based on the point-to-point forwarding paradigm, our work at least suggests that controlled flooding may offer a viable alternative to explicit route discovery and maintenance. One can think of numerous applications where the trivially low cost of nodes, simplicity, node scalability, and the completely automatic configurability outweigh the performance penalty. Among those applications are sensor networks, smart badges, profile matchers, and all those areas in which the networking component must be implemented in a tiny, disposable, and inconspicuous device.

Chapter 4

Power control for energy-efficiency

4.1 Introduction

Energy consumption control is one of the key design issues in ad-hoc wireless networks, with transmission power being the predominant factor in the overall energy budget. One natural formulation of the power control problem is choosing the minimum power level by each node, based on the position information of the reachable nodes, while maintaining global connectivity. Such a model assumes that the minimum power needed to reach a node depends solely on the distance to the node. This assumption implies the symmetry of the problem with respect to the two endpoints of a transmission path.

Consider an n-node, multi-hop, ad-hoc, wireless network deployed on a two-dimensional plane. Suppose that each node is capable of adjusting its transmission power up to a maximum denoted by P_{max}. Such a network can be modeled as a graph $G = (V, E)$, with the vertex set V representing the nodes, and the edge set defined as follows:

$$E = \{\langle x, y\rangle \mid \langle x, y\rangle \in V \times V \wedge d(x, y) \leq R_{max}\} ,$$

where $d(x, y)$ is the distance between nodes x and y and R_{max} is the maximum distance reachable by a transmission at the maximum power P_{max}. The graph G defined this way is called the *maximum powered network*. We also define the neighbour set $N(x)$ of the vertex x as

$$N(x) = \{y|(x, y) \in E\}$$

The degree of a given node x is the number of nodes in $N(x)$. The *density* of the graph is the average degree for each node.

The local choices regarding the transmission power at individual nodes will collectively shape a subgraph of the maximum powered network. The properties of that subgraph, e.g., its average degree, may have a significant impact on the performance of the routing layer. For example, flooding, used as a typical way of route discovery, may cause serious *broadcast storm* problems [39] in a dense graph, e.g., one close to the maximum powered network. By reducing the power level at each

node we also reduce the average node degree, which, in turn, reduces the contention in the node's neighborhood. Thus, it is generally beneficial to be able to broadcast route discovery messages over a proper subgraph of G.

The issue of selecting the optimum transmission power formulated in this context was first tackled by Roduplu et al. [52] who considered the so-called *enclosure graphs*. The local enclosure graphs constructed for individual nodes form globally a strongly connected graph guaranteed to contain the minimum-energy paths for all pairs of nodes. By applying to that graph a distributed Bellman-Ford algorithm with energy as the cost metric, one can find a minimum-energy end-to-end path for any pair of nodes.

We say that a graph $G' \subseteq G$ is a *minimum-energy path-preserving graph* or, alternatively, that it has the *minimum energy property*, if for any pair of nodes (u, v) that are connected in G, at least one of the (possibly multiple) minimum energy paths between u and v in G also belongs to G'. Minimum-energy path-preserving graphs were first defined in [28]. Typically, many minimum-energy path preserving graphs can be formed from the original graph G. It has been shown that the smallest of such subgraphs of G is the graph $G_{min} = (V, E_{min})$, where $(u, v) \in E_{min}$ iff there is no path of length greater than 1 from u to v that costs less energy than the energy required for a direct transmission between u and v.

Let $G_i = (V, E_i)$ be a subgraph of $G = (V, E)$ such that $(u, v) \in E_i$ iff $(u, v) \in E$ and there is no path of length i that requires less energy than the direct one-hop transmission between u and v. Then G_{min} can be formally defined as follows:

$$G_{min} = \bigcap_{i=2}^{n-1} G_i$$

It is easy to see that any subgraph G' of G has the *minimum-energy property* iff $G' \supseteq G_{min}$. Thereby, each of $G_i \supseteq G_{min}$, for any $i = 2, 3, \ldots, n-1$ is a *minimum-energy path preserving graph*.

Distributed construction of G_{min} by the nodes is somewhat tricky, and it contradicts its own goals, because it requires communicating with distant nodes using high power. On the other hand, graphs G_i can be built based on local information at considerably relaxed power requirements. An algorithm for constructing one such graph, G_2, was presented in [28]. It works reasonably well in dense networks but its performance degrades considerably when the network density drops to medium or low.

Note that while G_2 is only the first of the series G_i, it is the most interesting and most practically useful derivative of the maximum powered graph G. By increasing the value of i we try to account for longer and longer paths (with higher number of hops) that may turn out to require less energy than a direct hop. Because of the obvious facts that 1) the number of hops tends to directly correlate to distance, 2) the total transmission power of a path is additive on the number of hops, the likelihood of such paths drops rapidly once the case $i = 2$ has been handled. Consequently, considering that

the cost of discriminating among longer paths will unavoidably involve exchanging many messages, and thus will incur obvious energy overheads, it makes perfect sense to restrict our attention to G_2.

In this chapter, we show how to construct G_2 efficiently in sparse and moderately dense networks with some assistance of the Medium Access Control (MAC) layer. The proposed modifications affect the backoff procedure of the 802.11b collision avoidance scheme and are somewhat reminiscent of the previous work [12] on Quality of Service issues related to fairness and priority scheduling. In our own previous work [48], we proposed another modification to the collision avoidance mechanism of 802.11b aimed at improving the reliability of multicast transmissions in ad-hoc networks.

4.2 Minimum-energy path preserving graphs

4.2.1 Power model

We assume the well known, generic, channel path loss model, where the minimum transmission power is a function of distance [51]. To send a packet from node x to node y, separated by distance $d(x, y)$, the minimum necessary transmission power is approximated by

$$P_{trans}(x, y) = t \times d^{\alpha}(x, y) \ ,$$

where $\alpha \geq 2$ is the *path loss factor* and t is a constant. Signal reception is assumed to cost a fixed amount of power denoted by r. Thus, the total power required for one-hop transmission between x and y becomes

$$P_{total}(x, y) = t \times d^{\alpha}(x, y) + r$$

The model assumes that each node is aware of its own position with a reasonable accuracy, e.g., via a GPS device.

4.2.2 Previous approach to constructing G_2

The algorithm presented in [52] is based on the notion of *relay region*. Throughout the paper we will refer to that algorithm as R&M—for the sake of brevity. Given a node u and another node v within u's communication range (at P_{max}), the relay region of node v as perceived by u (with respect to u), $R_{u \to v}$, is the collection of points such that relaying through v to any point in $R_{u \to v}$ takes less energy than a direct transmission to that point (see Figure 4.1(a)).

Given the definition of relay region, the algorithm for constructing G_2 becomes straightforward. Suppose that u is the starting node of a path. If, as perceived by u, some node w falls in the relay region of some other node v, then w will not be included in the so-called *neighborset* of u (i.e., u will not transmit directly to w). Thus, by definition, the neighborset of node u will contain only those nodes that do not fall into relay regions of other nodes reachable by u. G_2 can be constructed by connecting each node with only those nodes that are included in its neighborset.

The efficiency of constructing G_2 using this approach depends on how inexpensively nodes can collect the position information of their neighbors. One trivial way to discover the position of all

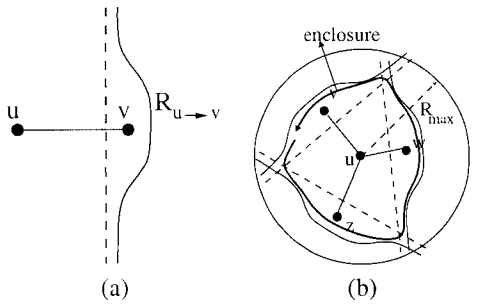


Figure 4.1: (a) The relay region of v with respect to u, (b) The enclosure of node u

nodes in the neighborhood is to periodically broadcast a *neighbor discovery message* (NDM) at the maximum power P_{max}, to which all reachable nodes will respond with their position information.

With power concerns in mind, it is natural to ask this question: "Is there a way for node u to restrict the search area to a subset of its transmission range?" Perhaps, in sufficiently many cases, u does not require the position information of all nodes that fall within its communication range to determine its neighborset. As it turns out, such confinement is often possible.

Given $R_{u \to v}$, the relay region of node v with respect to node u, the complement of this region, denoted by $R^c_{u \to v}$, is the set of points for which it is not power-efficient for u to use node v as a relay. Let $N(u)$ be the set of nodes that do not fall in the relay region of any other node in u's neighborhood. Then, $\bigcap_{k \in N(u)} R^c_{u \to k}$ is the set of points where u should transmit directly without using any relay. On the other hand, the direct transmission range of u is limited by P_{max}—the maximum transmission power. Let $F(u, P_{max})$ denote the circular region with radius R_{max} centered at u and describing its transmission range. The *enclosure* of node u is defined as the following set of points:

$$\epsilon_u = \bigcap_{k \in N(u)} R^c_{u \to k} \bigcap F(u, P_{max})$$

Figure 4.1(b) shows an example of enclosure. It is its enclosure beyond which a node need not search for neighbors. This observation lead in [28] to a power saving algorithm for constructing G_2. In a nutshell, instead of broadcasting the NDM at the maximum power, u will start with some initial power, $P_0 \ll P_{max}$. After collecting responses from the neighborhood, if the enclosure has been found, then there is no need to search any further. Otherwise u will re-broadcast the NDM at an increased power level and try again. This process will continue until u either finds the enclosure or reaches P_{max}. Figure 4.2 gives a high level description of that algorithm, which we shall refer to as the *Reduced Neighbor Search Algorithm*, or RNSA for short. Note that the efficiency of RNSA depends on the number of iterations required to find the enclosure, which in turn depends on the initial power P_0 and the power increment P_{inc} applied in step 6.

Algorithm RNSA:

1. *transmission power* := P_0
2. loop
3. Broadcast NDM and collect responses.
4. Update the neighborset using the definition of relay region.
5. If enclosure found or *transmission power* = P_{max} then exit.
6. Increase *transmission power* by P_{inc}.
7. endloop

Figure 4.2: Reduced Neighbor Search Algorithm (RNSA)

4.2.3 Problems with RNSA

RNSA suffers from two major drawbacks. First, while the algorithm works fine when the network is dense, in a sparse or moderately populated network it tends to exhibit poor performance. To see the reason for this, let us note that the enclosure of a node u can be formed in one of two possible ways:

Case (i): the enclosure is determined solely by the nodes in $N(u)$. This happens when the following condition holds:

$$\bigcap_{k \in N(u)} R^c_{u \to k} \subseteq F(u, P_{max})$$

Such an enclosure is the intersection of the complements of the relay regions of all nodes in $N(u)$ (see Figure 4.1(b)). We will call it an *enclosure by neighbors*.

Case (ii): the transmission range of u is a limiting factor, i.e.,

$$\bigcap_{k \in N(u)} R^c_{u \to k} \neq \bigcap_{k \in N(u)} R^c_{u \to k} \bigcap F(u, P_{max})$$

Such enclosures are called *enclosures by maximum boundary* (see Figure 4.3).

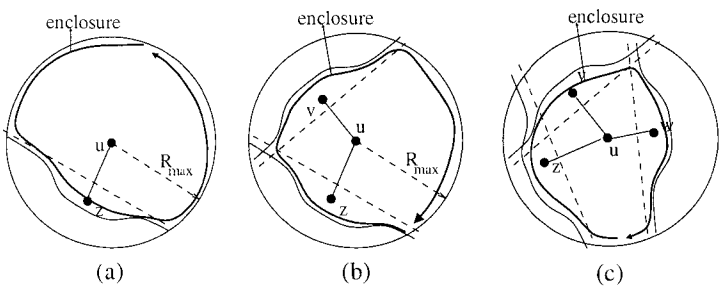


Figure 4.3: Enclosures by maximum boundary

If a node has an enclosure by neighbors, then, in principle, it need not transmit at P_{max} to find that enclosure. For such nodes, RNSA is useful and may bring about power savings compared to the naive scheme. On the other hand, a node having an enclosure by maximum boundary, will ultimately need to search with maximum power. For such a node, RNSA performs worse than the

naive scheme as it runs through a number of essentially futile iterations before reaching P_{max}. The traffic caused by the NDMs broadcast during those iterations and the multiple repetitive responses to those messages wastes bandwidth and contributes to the noise in the neighborhood.

One can naturally expect that the likelihood of finding a node whose enclosure is determined by neighbors is higher in a dense network and at locations further from the network's edge. On the other hand, sparse networks will have many nodes with maximum boundary enclosures.

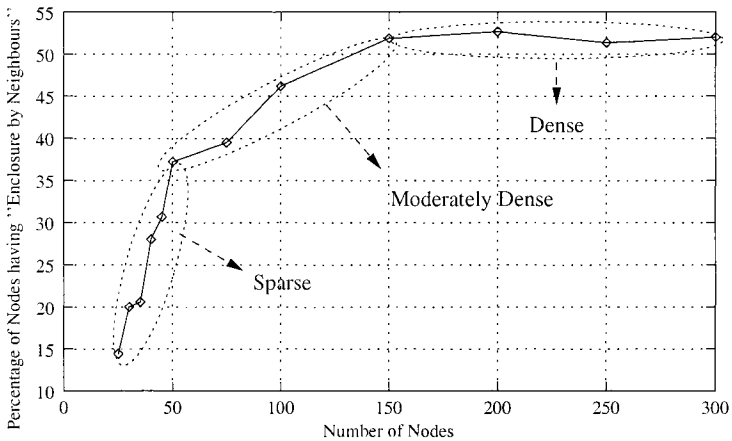


Figure 4.4: Percentage of nodes with enclosures by neighbors

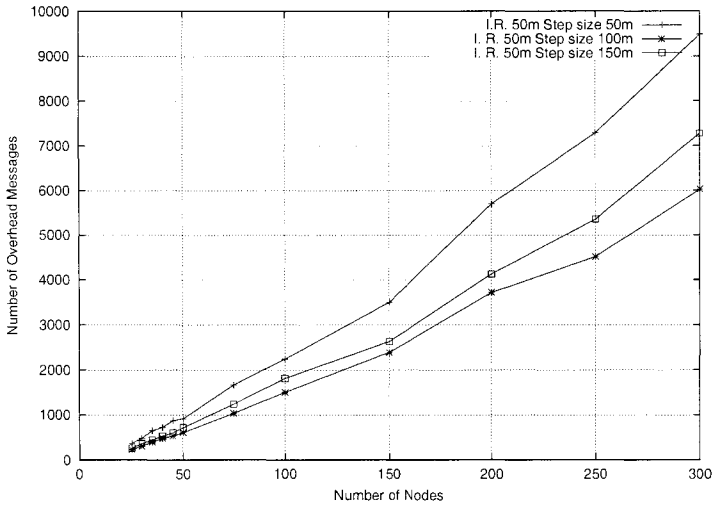
Figure 4.4 shows some statistics relating the observed percentage of nodes with enclosures by neighbors to the network density. The density of the network in this experiment was determined by the total number of nodes, which were distributed uniformly in a fixed square region of $670m \times 670m$. Each point was obtained as the average of 5 distribution samples.

The percentage of nodes having enclosures by neighbors is between 14 and 38% when the total number of nodes is less than 50 (sparse network), between 38 and 52% for $50 - 150$ nodes (moderately dense network), and greater than 52% for the total number of nodes greater than 150 (dense network). This picture clearly suggests that RNSA will not perform well for sparse or moderately dense networks, where most nodes have to transmit at P_{max} to find their enclosures.

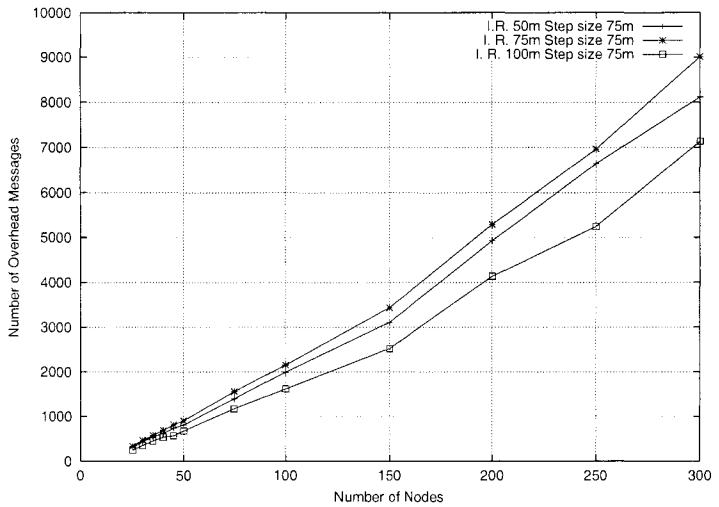
The second problem with RNSA is the lack of clear guidelines regarding the selection of initial power P_0 and the increment P_{inc}. Figure 4.5, showing the relationship between those parameters and the resulting message overhead of RNSA, demonstrates that their choice is not irrelevant.

For simplicity, the power level shown in Figure 4.5 has been transformed into the transmission range (see Section 4.2.1). In part (a), the initial transmission range is the same for all three curves ($50m$), but the increments are different: $50m$, $100m$, and $150m$. Especially for dense networks, where RNSA is most useful, the differences are considerable and exceed 50%.

In part (b), the step size is fixed at $75m$, while the initial transmission range varies between $50m$ and $100m$. Again, the selection of P_0 affects the observed overhead quite significantly.



(a)



(b)

Figure 4.5: Comparing number of overhead messages by varying (a) Step size, (b) Initial communication range

Of course, the simple exercise illustrated in Figure 4.5 does not allow us to draw quantitative conclusions regarding the recommended setting of the two parameters of RNSA. As the observed susceptibility of the algorithm's performance to those parameters is rather high, one can expect that their optimum setting is also highly sensitive to various characteristics of the network. As those characteristics in ad-hoc networks tend to be diverse and often dynamic, there is little hope that the algorithm can dynamically adapt itself to offer its best performance in every possible configuration.

4.3 Constructing G_2 for sparse and moderately dense networks

In this section we describe an algorithm for constructing G_2 that works more efficiently than RNSA in sparse and moderately dense networks. Our algorithm is named BICOMP, for *BIased COntention at Maximum Power*. We shall start by defining some terms.

4.3.1 Cover region and cover set

Consider a pair of nodes (s, f), such that f lies within the communication range of s, i.e., is reachable by s at P_{max}. Envision the set of all points that can possibly act as relays between s and f, such that it would be more power efficient for s to use an intermediate node located at one of those points instead of sending directly to f. Note that, owing to the symmetry of our underlying propagation model, exactly the same set is defined by considering f as the starting point. We will call it the *cover region* of s and f and denote by $C_{(s,f)}$. The collection of all nodes falling into the cover region of s and f will be called the cover set of s and f. Formally the cover region and cover set, are described by the following definition.

Definition 1: The cover region $C_{(s,f)}$ of a pair of nodes (s, f), where f is reachable from s, is defined as:

$$C_{(s,f)} = \{<x, y> \mid td^{\alpha}(s, <x, y>) + td^{\alpha}(<x, y>, f) + r \leq td^{\alpha}(s, f)\} ,$$

where $\alpha \geq 2$. In the above equation, $d(s, <x, y>)$ denotes the distance between node s, and a hypothetical node located at $<x, y>$, and r is the fixed receiving power. The cover set of the same pair (s, f) is

$$\xi_{(s,f)} = \{v \mid v \in V \wedge Loc(v) \in C_{(s,f)}\}$$

Figure 4.6 shows two examples of cover regions, with the path loss exponent $\alpha = 2$ and $r = 0mW$, and $\alpha = 4$, $r = 20mW$.

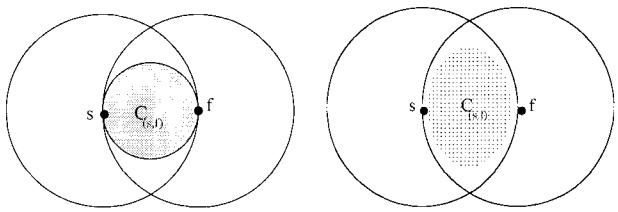
The following Lemma introduces two useful properties of cover regions:

Lemma 1 : (a) For any $c \in \xi_{(s,f)}$, $d_{sc} < d_{sf}$, (b) If $c \in \xi_{(s,f)}$ then $f \notin \xi_{(s,c)}$.

Proof : (a) If $c \in \xi_{(s,f)}$ then from definition 1 it follows that, $d_{sc}^{\alpha} + d_{cf}^{\alpha} + r/t \leq d_{sf}^{\alpha}$. Now, for $r > 0$ and $\alpha \geq 1$, $d_{sf} > d_{sc}$.

(b) If $c \in \xi_{(s,f)}$, then from (a) $d_{sf} > d_{sc}$. Now suppose that also $f \in \xi_{(s,c)}$ then again from (a), $d_{sc} > d_{sf}$ which is a contradiction.

Note the difference between relay regions and cover regions. Given a node pair (u, v), the relay region provides an answer to this question: "What are the points for which node v can act as a power-efficient relay for node u?" On the other hand, the question answered by the cover region is: "What are the points that can act as power-efficient relays for node u when sending to v?" These questions are quite different; in particular, cover regions are indifferent to the ordering of u and v, while relay regions are not.



(a) (b)

Figure 4.6: Cover regions: (a) $\alpha = 2, r = 0mW$ (b) $\alpha = 4, r = 20mW$

4.3.2 Constructing G_2

As in RNSA, the operation is described from the viewpoint of one node s. In contrast to RNSA, s broadcasts a single neighbor discovery message (NDM) at the maximum power P_{max}. For now, let us assume that all nodes receiving the NDM from s send back a reply. Later we will explain how the number of replies can be reduced with the assistance of the MAC layer. The reduced overhead in our algorithm will result from the reduced number of replies in a dense network. As it turns out, those savings outweigh the gains of the reduced power transmission of the NDM in RNSA, especially in networks that are not very dense.

While s collects the replies of its neighbors, it learns their identities and locations. It also constructs the cover sets of those neighbors. Initially, all those sets are empty (s does not even know what neighbors there are). The set A_s, which also starts empty, keeps track of all the nodes discovered in the neighborhood.

Whenever s receives a reply to its NDM from a node v, it performs the algorithm listed in Figure 4.7. Its purpose is to update the configuration of the cover sets. At the end, when s has received all the replies, the configuration of cover sets is complete.

updateCoverRegion(s, v)
begin
  for each $w \in A_s$
    if $Loc(v) \in C_{(s,w)}$ *then*
      $\xi_{(s,w)} = \xi_{(s,w)} \cup \{v\}$;
    else if $Loc(w) \in C_{(s,v)}$ *then*
      $\xi_{(s,v)} = \xi_{(s,v)} \cup \{w\}$;
  $A_s = A_s \cup \{v\}$;
end

Figure 4.7: Algorithm for building cover sets

The goal of node s is to determine its *neighborset*, i.e., the collection of neighbors to which transmission should be direct. Having determined the cover regions of all its neighbors, s is in position to identify the members of its neighborset. If $\xi_{(s,v)} \neq \emptyset$ for some v, it means that sending directly to node v is not power efficient: there is at least one node $w \in \xi_{(s,v)}$ that can act as a power-efficient relay between s and v. On the other hand, a node v that has an empty cover set with s, but belongs

to the neighborhood of s, i.e., is present in A_s, necessarily has no power-efficient relays and thus belongs to the neighborset of s. Consequently, the loop listed in Figure 4.8 completes the algorithm by generating the neighborset of s denoted by $\aleph_s$.

```
neighbor(s)
begin
    ℵ_s = ∅
    for each v ∈ A_s
        if ξ_(s,v) = ∅ then
            ℵ_s = ℵ_s ∪ {v};
end
```

Figure 4.8: Generating the neighborset of s

Example: We will demonstrate the process of constructing G_2 with an example. Figure 4.9(a) shows the simple scenario used in this example. Here, node S has four neighbors a, b, c, and d within its maximum transmission range. Before node S runs the algorithm, it needs to initialize its two sets A_s and $\aleph_s$ to $\emptyset$ sets. In the process of finding neighbors in G_2, node S broadcasts a neighbor discovery message (NDM). All four nodes receiving this NDM, are supposed to send back reply messages appending their current position information. The reply messages will arrive at node S sequentially one after another. Let us assume that a's reply arrives first. This is shown in Figure 4.9(c). Upon receiving a's reply, node S runs the "*updateCoverRegion*" algorithm mentioned in Figure 4.7 and then runs the algorithm in Figure 4.8 to generate the neighborset $\aleph_s$.

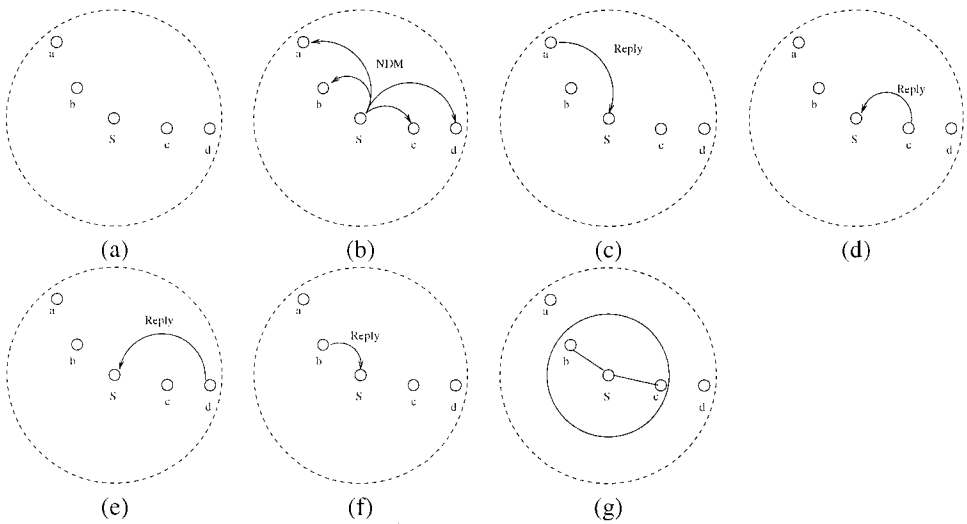


Figure 4.9: Node S's construction process for its neighborset $\aleph_s$ in G_2. (a) The simple scenario used in the example. Node S initially has four nodes a, b, c and d as neighbor. (b) Node S broadcasts a neighbor discovery message (NDM). (c)-(f) Node a, b, c and d reply one after another appending their position information. (g) Node S's final neighborset is, $\aleph_s = \{b, c\}$. So it reduces its transmission range just to cover node b and c only. The reduced transmission range is shown by the circle in dark line.

The coverset $\xi_{(S,a)}$ is currently $\emptyset$ because node b has not sent its reply yet. The updated neighborset $\aleph_s$ is $\{a\}$. This is shown in the second row of Table 4.1. After a while, node S eventually receives other reply messages. Let us assume that the reply messages come in the following order: (i) node c's reply, (ii) node d's reply and finally, (iii) node b's reply. This is shown in Figure 4.9(d), (e) and (f) respectively. For every reply message node S will do similar things, run two algorithms in Figure 4.7 and Figure 4.8. The result of running these two algorithms and the corresponding changes in neighborset $\aleph_s$ is shown in third, fourth and fifth rows of Table 4.1. After receiving all the reply messages, the final neighborset becomes $\aleph_s = \{b, c\}$, which means node S is free to exclude node a and d from its transmission range and reduce the range in such a way so that only node b and c are just covered. The final transmission range chosen by node S is shown in Figure 4.9(g).

ILLUSTRATING CONSTRUCTION PROCESS OF G_2

Event	Neighborset in G_{max}	Coverset	Neighborset in G_2
Initially	$A_s = \emptyset$		$\aleph_s = \emptyset$
Reply from a	$A_s = \{a\}$	$\xi_{(S,a)} = \emptyset$	$\aleph_s = \{a\}$
Reply from c	$A_s = \{a, c\}$	$\xi_{(S,c)} = \emptyset$	$\aleph_s = \{a, c\}$
Reply from d	$A_s = \{a, c, d\}$	$\xi_{(S,d)} = \{c\}$	$\aleph_s = \{a, c\}$
Reply from b	$A_s = \{a, c, d, b\}$	$\xi_{(S,a)} = \{b\}$	$\aleph_s = \{b, c\}$
		$\xi_{(S,b)} = \emptyset$	

Table 4.1: The construction process of node S's neighborset in G_2.

4.3.3 BICOMP: reducing the number of reply messages

The primary advantage of BICOMP is that it is able to reduce the number of reply messages, and thus significantly lower the overall power expense needed to discover the resultant graph. Consider a simple scenario where s can reach only two nodes v and w within the radius of maximum transmission range, such that $v \in \xi_{(s,w)}$. Clearly, from Lemma 1(b), $w \notin \xi_{(s,v)}$. The neighborset of s, computed by the algorithm in Figures 4.7 and 4.8, $\aleph_s = \{v\}$. When node s broadcasts its neighbor discovery message, nodes v and w are supposed to send back a reply message with their location information. Both nodes v and w, will contend for access to the shared wireless channel to send their reply messages back to s. If v wins, then the reply of w will be received by s after the message sent by v. Note that the message sent by w will be redundant: it will not affect the outcome of the algorithm, as w is covered by v and it should not be included in $\aleph_u$. On the other hand, if w wins and sends its reply first, the algorithm will first add w to A_u and then, after receiving the second message from v, add v to $\xi_{(s,w)}$.

Note that if v were allowed to win, and w overheard the reply of v, then w could refrain from sending its reply to s. Node w is in the same position to find out that its message is redundant as node s: it has the location information of node s (which arrived in the NDM of s) and can carry out exactly the same simple calculations as node s. This way, some replies can be eliminated before

being transmitted.

In a general scenario, we would like to be able to enforce some ordering of the reply messages that would give precedence to those likely to be relevant and postpone those likely to be redundant. A node detecting that its pending message is redundant would drop it and thus reduce the amount of traffic needed for neighborset discovery.

To be able to order the reply messages, we need to exercise some control over the contention resolution mechanism used in the MAC layer. With IEEE 802.11b, a node willing to transmit a packet under contention has to wait for a certain number of idle slots chosen at random in the range of $[0, cw - 1]$, where cw is the so-called *contention window*. Statistically, different nodes are likely to pick different numbers, which will help them transmit without interference in different time slots. To influence the order of transmission in a way that would be compatible with our sense of relevance of the reply messages, we have to bias the random distribution of the slot selection process.

Note that generally we cannot eliminate randomness from the process. Whatever idea a node may have regarding the selection of its transmission slot, the decision is always local and thus cannot preclude other nodes from arriving at the same decision. This may happen when two or more nodes find themselves in the same (or similar) situation with respect to s and conclude that their priorities are the same. To avoid permanent lockouts in such situations, the contention resolution scheme must not give up its random component.

Our intention is to make the expected waiting time (the number of skipped slots) an increasing function of the distance from the node that sent the NDM. This will increase the chance that covered nodes will schedule their transmissions later and, consequently, the chance that those transmissions will never take place. According to Lemma 1(a), if a node v is in the cover set of node w, then d_{sv} must be less then d_{sw}. A natural way to proceed is to divide the area around node s into partitions according to the distance from s.

4.3.3.1 Equal-area partitions

Let $F(s, P_{max})$ represent the circular region of radius R_{max} reachable by s at its maximum transmission power. We divide $F(s, P_{max})$ into n equal-area partitions. A node v is said to fall into partition i, $1 \leq i \leq n$ iff,

$$\begin{cases} 0 < d_{sv} \leq R_{max} \times \sqrt{\frac{i}{n}} & \text{when } i = 1 \\ R_{max} \times \sqrt{\frac{i-1}{n}} < d_{sv} \leq R_{max} \times \sqrt{\frac{i}{n}} & \text{when } i = 2 \ldots n \end{cases}$$

Note that, in this scheme, the circle $F(s, P_{max})$ centered at node s is divided into n partitions, all with the same area of

$$A = \frac{\pi R_{max}^2}{n}$$

One can argue that partitioning nodes this way makes sense because, assuming the uniform distribution of nodes, every partition will tend to contain about the same number of nodes. The issue is

far from being that simple, however. This is because nodes located closer to s are more likely to participate in the neighborset. Consequently, it may be sensible to group more distant nodes into larger classes, providing for finer contention resolution in a closer neighborhood of s.

4.3.3.2 Equal-width partitions

With this scheme, $F(s, P_{max})$ is divided into n equal-width partitions. A node v is said to fall into partition i, $1 \leq i \leq n$ iff,

$$\frac{R_{max} \times (i-1)}{n} < d_{sv} \leq \frac{R_{max} \times i}{n}$$

This time, the circle $F(s, P_{max})$ centered at node s is divided into n circular strips with the same width of R_{max}/n. The area of partition i is

$$A_i = \frac{\pi R_{max}^2 (2i-1)}{n^2}$$

and it increases with the partition radius.

4.3.3.3 Modifying the backoff mechanism

A node falling into partition i chooses a random number prescribed by the following formula:

$$R = (i-1) \times \frac{cw}{2^{\lceil log_2 n \rceil}} + \lfloor U(0,1) \times \frac{cw}{2^{\lceil log_2 n \rceil}} \rfloor \ ,$$

where $U(0,1)$ is a uniform distribution between 0 and 1. If n is a power of 2, the equation becomes a bit simpler.

Example: Let $n = 2$. The transmission range of s is divided into two partitions. According to the equal-width scheme, the nodes whose distance from s is less than $R_{max}/2$ are be assigned to partition number 1, and all the remaining nodes fall into partition 2. Assuming the contention window size $cw = 32$, the nodes in partition 1 choose a random number between 0 and 15 and the nodes in partition 2 select a random number between 16 and 31.

4.3.4 Extraneous edges

Consider the simple scenario shown in Figure 4.10(a). Suppose s is sending an NDM message. In this configuration, $u \in \xi_{(s,v)}$ (and also $v \in \xi_{(s,w)}$), but $u \notin \xi_{(s,w)}$. Both RNSA and R&M do not depend on the ordering of reply messages. The final graph produced by RNSA and R&M is shown in Figure 4.10(b) and (c) respectively. Note that the graph constructed by R&M has one edge more than the one produced by RNSA. The shape of the final graph found by our algorithm depends on the order of reply messages. If u sends its reply before v, then v will cancel its reply because it is covered by u. Later on, when w sends its reply, s will not be able to detect that w is covered by v (s does not know v's position because v has canceled its message) and will add an extra edge between s and w, similar to R&M.

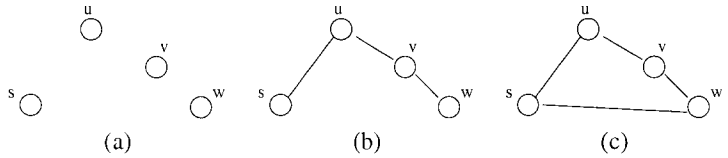


Figure 4.10: An extraneous edge

Now, if v sends its reply before u, then the edge between s and w will not show up in the final graph (because s will know the position of v). Thus, in that case, our algorithm will produce the same graph as RNSA. In other words, the exact outcome of our algorithm depends to a certain extent on the ordering of reply messages. In the best case, the algorithm will produce exactly G_2, in the worst case it will generate a graph similar to the output of R&M, and on the average it will produce something in between.

Note that there is a relatively easy way to modify BICOMP to avoid inserting the extraneous edges, and to produce exactly G_2. For that, we need to be able to reschedule some of the previously canceled messages. For example, in the scenario shown in Figure 4.10, having canceled the reply message and then overhearing the reply sent by w, node v may reschedule its reply to notify s about its location. Of course, this will result in an increased number of reply messages (and thus a higher power overhead), especially in scenarios more complex than the one illustrated in Figure 4.10.

On the other hand, the likelihood of the extraneous edges diminishes in situations where the problematic node is covered by multiple nodes (as is likely in many practical scenarios). Although one of those covering nodes may cancel its reply, there exist other covering nodes whose replies may make it to s, which will then be able to eliminate the extra edge. The probability that all covering nodes will cancel their replies may turn out to be sufficiently low to be acceptable.

RESULTANT GRAPH SIZE

Number of nodes	Edges in G_2	Edges in LOHG	Edges in R&M
25	68	68	70
30	78	80	82
35	104	110	112
40	142	148	148
45	182	188	192
50	184	188	192
75	352	362	368
100	622	634	638
150	1244	1248	1248

Table 4.2: Number of edges in the resultant graph

We have carried out experiments to assess the magnitude of the problem, i.e., find out how many extraneous edges tend to be included by BICOMP. The results show that the percentage of those edges is quite low. For illustration, Table 4.2 shows a comparison between the number of

edges in G_2, those constructed by BICOMP (as described in Section 4.3.3), and those found by R&M. Note that, owing to the random character of contention resolution, the numbers for BICOMP reflect one of several possible outcomes, which are always bound from above by the last column. The possible reduction in the number of edges—acquired by complicating BICOMP to reschedule some of the dropped replies—appear to be insignificant, and they do not warrant the added power expense. Consequently, we have decided to ignore the issue and not to reschedule any canceled reply messages. The final topology produced by our algorithm, which may be slightly larger than G_2, will be called a *Low OverHead Graph*, or LOHG for short. One can easily see that the following Lemma holds.

Lemma 2 : (a) LOHG contains G_2. (b) LOHG is connected.

Proof : (a) According to our algorithm each node u starts the process by broadcasting an NDM message. Suppose the set of nodes receiving this NDM message is $N(u)$. Each node v receiving the NDM message will initially schedule a reply but some of those nodes will cancel if they overhear a reply message from any of its covering node. Let $N_2(u)$ denote u's neighbor in graph G_2. Now, for every node $w \in N_2(u)$ it is true that $\xi_{(u,w)} = \emptyset$. In other words, there is no covering node for w. Hence, every node $w \in N_2(u)$ will never cancel its reply message. As a result, all nodes in $N_2(u)$ will be included in the final neighborset of u. Therefore the *Low Over-Head Graph* will contain G_2.

(b) At first we will show that G_2 is connected. Let us define the cost of an edge $< u, v >$ denoted by $p(u, v)$ equals to the required transmission power to send any message from u to v. [1] A path between two nodes consists of zero or more intermediate nodes. Let the cost of a path $r = (v_0, v_1, ...v_n)$ of length n denoted by $C(r)$, be expressed as:

$$C(r) = \sum_{i=0}^{k-1} p(v_i, v_{i+1})$$

Consider a minimum energy path r between a pair of node (x, y) in the original graph G. Without loss of generality, we can assume that r is the longest among all minimal-energy paths between (x, y). Suppose $r = (u_0, u_1, ..., u_n)$. Now, for all $i = 0, 1, 2...n - 1$ it will happen that $(u_i, u_{i+1}) \in G_2$. Otherwise, suppose $(u_k, u_{k+1}) \notin G_2$. Then there exists a path r_k of length 2 such that $C(r_k) < p(u_k, u_{k+1})$, that's why (u_k, u_{k+1}) is excluded from G_2. But then it is immediate that replacing (u_k, u_{k+1}) by r_k in the path r will create another longer path r' such that $C(r') < C(r)$ contradicting the choice of r.

From above paragraph, it is clear that G_2 will preserve all minimum-energy paths between any pair of nodes, hence G_2 is connected. As G_2 is connected and LOHG contains G_2, it follows directly that LOHG is also connected.

[1] Due to symmetricity of power law the cost will be the same in the reverse direction

4.4 Experimental results

To evaluate the performance of our algorithm, we used a detailed simulation model based on *ns-2* [1] with wireless extensions. The distributed coordination function (DCF) of the IEEE standard 802.11 [13], was used as the MAC layer. The radio model characteristics were similar to Lucent's WaveLAN [62].

Initially, we deployed $25 - 300$ nodes over a flat square area of $670m \times 670m$. Two kinds of node distributions were used. The first one is *uniform distribution* and the second one is *Zipf-like distribution*. The performance of our algorithm will be discussed next in two separate subsections based on these two node distributions.

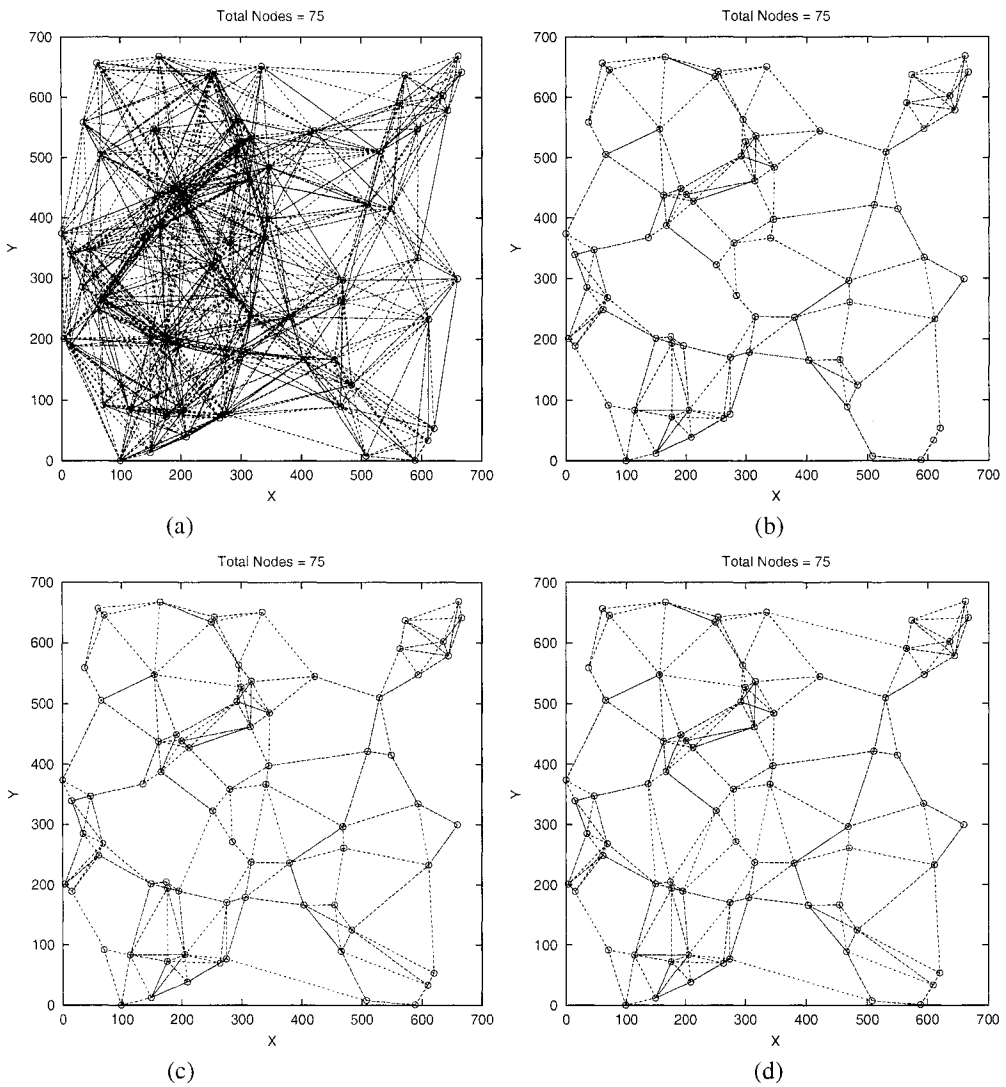
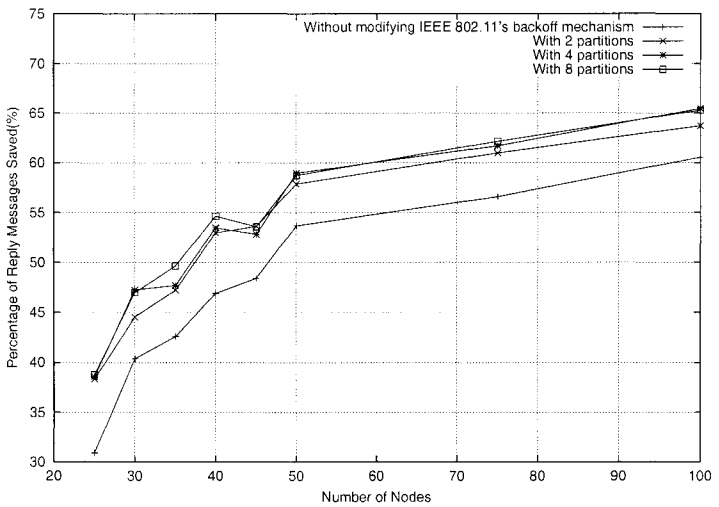


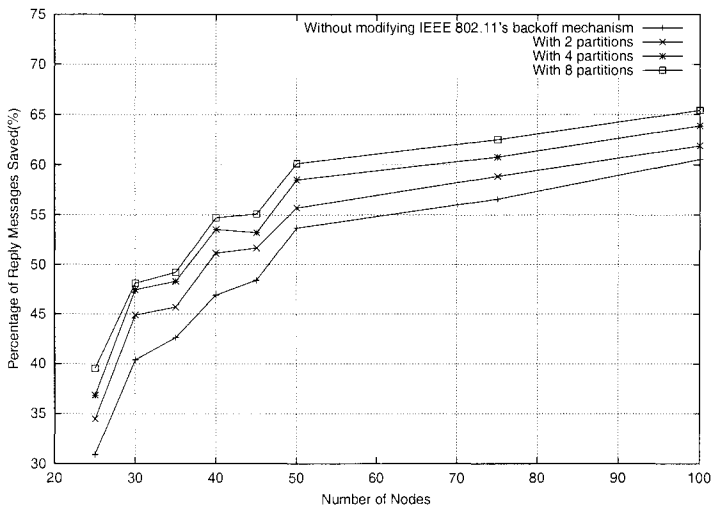
Figure 4.11: Subgraphs obtained by different algorithms: (a) Original graph, (b) G_2 generated by RNSA, (c) LOHG, and (d) Generated by R&M.

4.4.1 Uniform distribution of nodes

We define the *coverage area* of a node as the area around the node where the received signal power of transmitted signal from this node is above than a certain threshold[2] γ. According to the generic path-loss model of power attenuation—power exponentially decays over distance. As a result the coverage area becomes circular.



(a)



(b)

Figure 4.12: Savings versus number of nodes (a) With Equal-length Partition and (b) With Equal-Area Partition

Our first model assumes a uniform distribution of nodes over a two-dimensional area with finite size, but definitely larger than the coverage area. With uniform distribution, any position within this two-dimensional region is equally probable to be occupied by a node. Figure 4.11(a) shows a

[2]value of γ depends on receiver's sensitivity

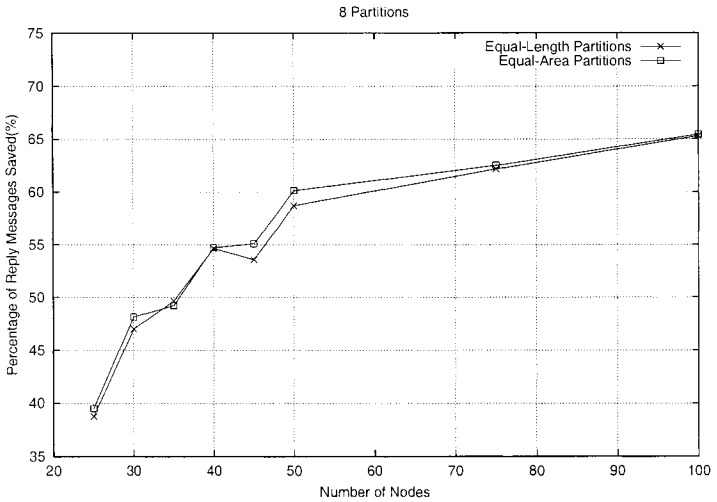


Figure 4.13: Comparing two partitioning scheme

typical deployment of 75 nodes, each node having a maximum communication range of $250m$. This is the starting graph G for our algorithm. The remaining parts of Figure 4.11 show the subgraph G_2, LOHG, and the graph found by R&M. Needless to say, all three subgraphs have considerably fewer links and a lower average node degree than the original maximum powered graph. LOHG has only 2.84% more links than G_2, while the R&M graph has 4.26% more links than G_2.

We ran experiments to see the effect of the varying partition size on the performance of BICOMP, specifically the ability of the biased backoff function to assist the algorithm in prioritizing the reply messages. The performance metrics of interest was the *Saving Ratio* defined as follows:

$$\text{Saving Ratio} = \frac{N_{cancel}}{N_{sent} + N_{cancel} + N_{dropped}} \times 100(\%) \ ,$$

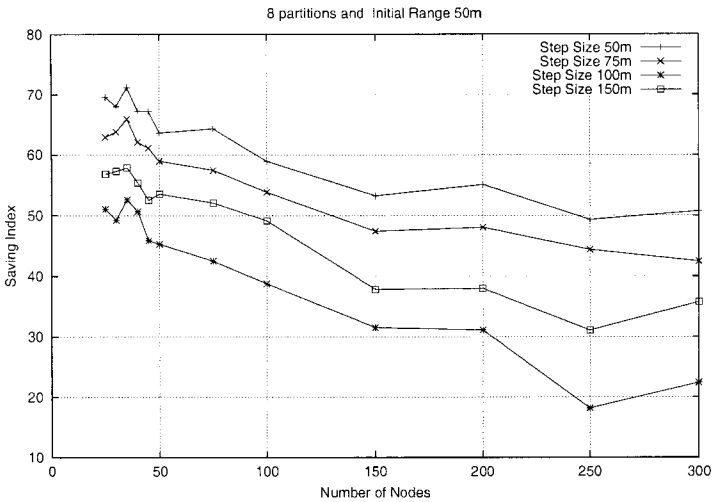
where N_{sent} is the total number of reply messages sent for each NDM requests, N_{cancel} is the number of messages that have been canceled because they were found redundant before transmission, and $N_{dropped}$ is the number of packets dropped by the MAC layer.[3]

Figure 4.12 shows the *Saving Ratio* for different node density. Each point in the figure is the average from 5 experiments using different samples with the same number of nodes. Three different numbers of partitions, 2, 4 and 8, were considered. Performance with the standard backoff mechanism is also included for reference.

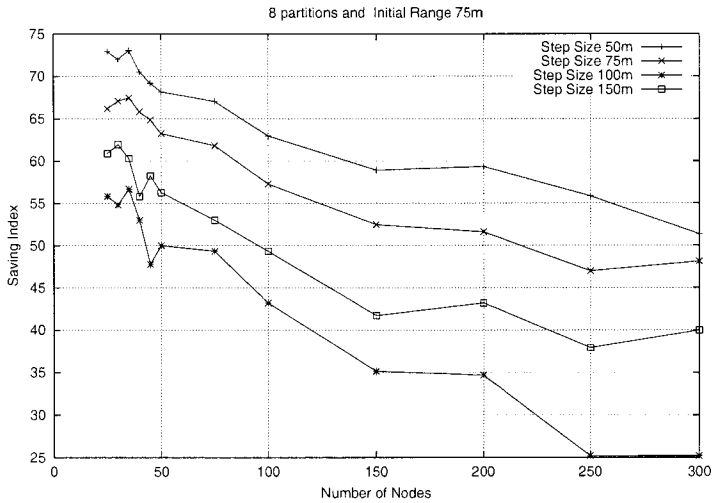
The savings appear to be considerably higher for denser networks. With the proposed MAC modification in which a node can overhear other nodes' reply messages, a higher density allows a node to cancel more messages than in a low density environment. Finer partitions also tend to exhibit slightly better performance. The total number of canceled replies was between 30 and 68%.

Figure 4.13 compares the two partitioning schemes. The *Equal-Area* partitioning seems to

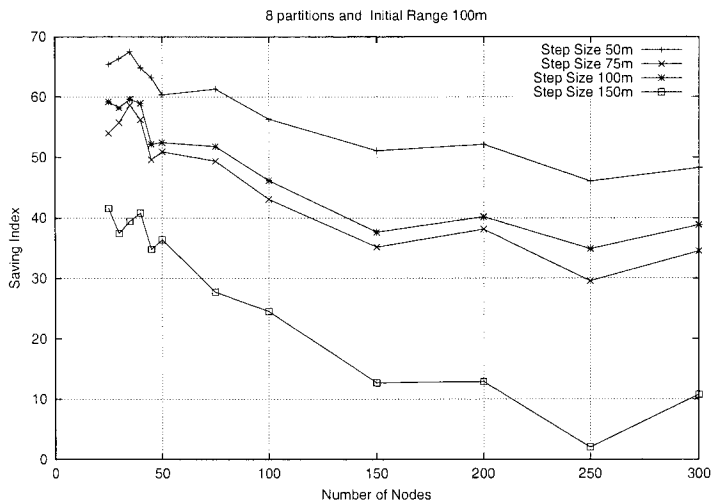
[3] According to IEEE 802.11, after a certain number of unsuccessful transmission attempts the MAC layer drops the packet.



(a)



(b)



(c)

Figure 4.14: BICOMP versus RNSA

slightly outperform the other scheme.

Figure 4.14 compares the performance of BICOMP with RNSA. The *Saving Index* is defined as follows:

$$\text{Saving Index} = \frac{N_{RNSA} - N}{N_{RNSA}} \times 100(\%) \ ,$$

where N is the total number of reply messages needed by our algorithm to construct LOHG, and N_{RNSA} is the corresponding number of messages observed in RNSA. The different graphs correspond to different initial transmission range settings for RNSA (translated from the initial transmission power), and the different curves in each graph correspond to different power increments. The number of partitions used by BICOMP was 8 in all cases.

For sparse networks, the *Saving Index* is very high and drops with the increasing density of nodes. In particular, $35-75\%$ of reply messages were eliminated with our approach for less then 50 nodes, and $25-55\%$ savings were observed for networks between 50 and 100 nodes. In some cases the Saving Index became close to zero. This means that under favorable conditions (if the network is dense enough), RNSA can perform well, but in a sparse or moderately dense network, our approach always brings about a significant improvement.

4.4.2 Zipf-like distribution

Finally, we experimented with a Zipf-like distribution of nodes, to see what happens when the network layout departs from uniformity. This study is important because the benefits of the bias in resolving contention during the discovery phase of BICOMP have been argued assuming a more or less balanced structure of a node's neighborhood.

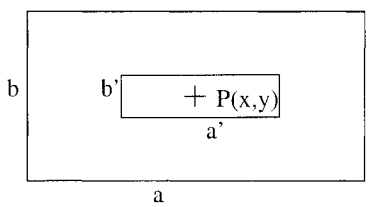


Figure 4.15: Zipf Distribution

Following the commonly used Zipf bias, we assumed that 80% of nodes are distributed over 20% of total deployment area and the remaining 20% nodes are distributed over the remaining 80% of the deployment area. For a more formal description, consider Figure 4.15. The total deployment area is the larger rectangular region with the dimensions a and b. Let P (the focal point of the distribution) be located at (x, y). The smaller rectangle centered at P has dimensions a' and b' such that

$$\frac{a'b'}{ab} = \alpha \ ,$$

where $\alpha = 0.2$. The network was generated in such a way that the probability of a node falling within the interior rectangle was $\beta = 1 - \alpha$, i.e., 0.8 in our case.

Figure 4.16 shows a typical topology reduction scenario involving 75 nodes under this biased distribution of nodes. The maximum communication range of each node was $250m$.

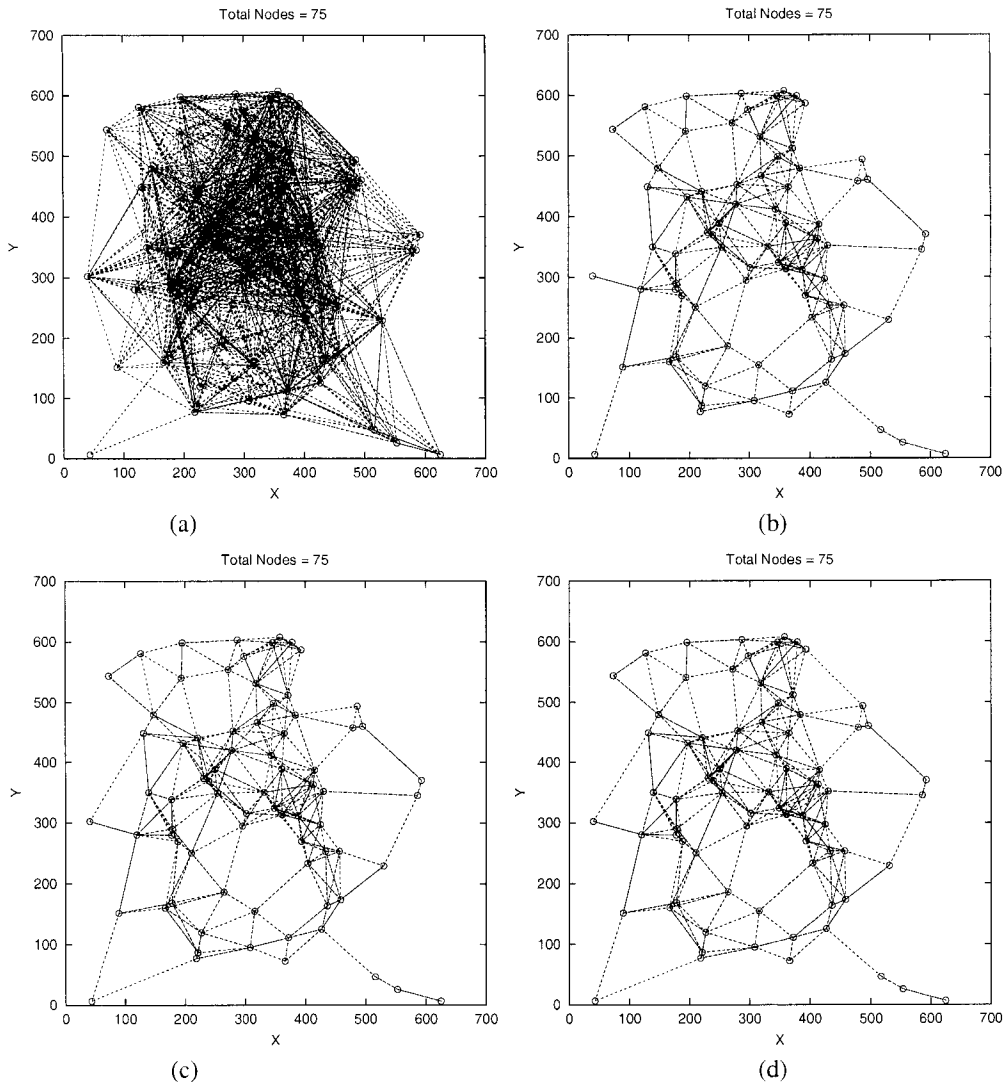
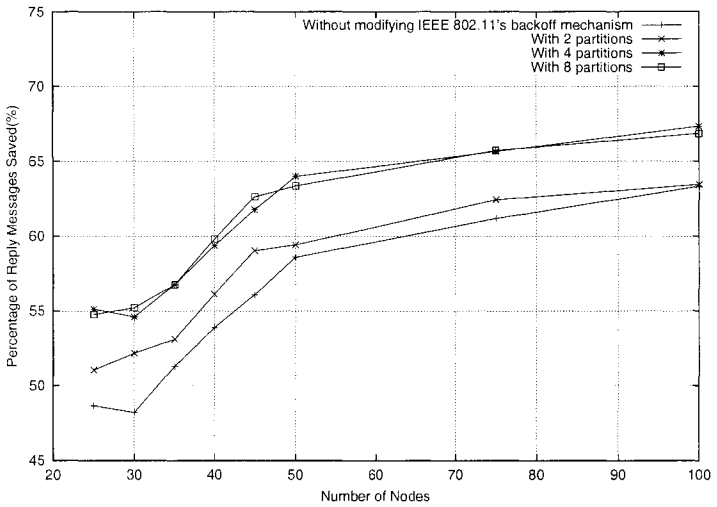


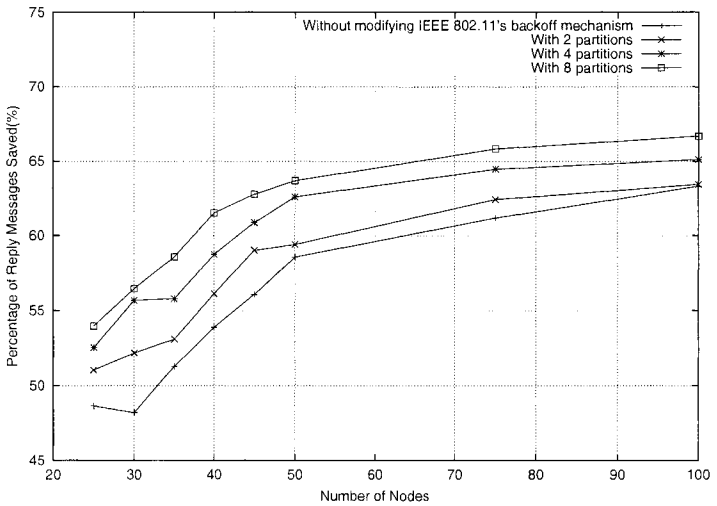
Figure 4.16: Reduced graphs under Zipf distribution of nodes: (a) Original graph, (b) G_2 generated by RNSA, (c) LOHG, and (d) Generated by R&M.

In Figure 4.17, we show the observed Saving Ratio with different node density under Zipf distribution. Again, three different numbers of partitions, 2, 4 and 8, were considered.

Note that this time the savings of BICOMP for sparse networks are even higher. This seems to dispel our worries that biased distributions may be detrimental to the advantages of biased contention.



(a)



(b)

Figure 4.17: Savings of BICOMP under Zipf distribution (a) With Equal-length Partition and (b) With Equal-Area Partition

4.5 Chapter summary

We have presented a MAC-assisted algorithm for constructing minimum energy path preserving graphs in ad-hoc wireless networks. Our studies have demonstrated the superiority of the new algorithm over the previous solution for networks with moderate and low density of nodes.

The notion of minimum-energy path preserving graphs is important from the viewpoint of network performance, even if we completely ignore the power saving gains. It gives a natural and rational way of reducing the degree of the network graph, which allows the nodes to reduce the number of neighbors they have to talk to and thus reduce the overall contention to the scarce radio

channel.

Our exercise demonstrates once again that strict protocol layering is a curse of wireless networks. The issue of power control calls for the collaboration of all layers and keeping some layers closed may significantly impair the flexibility of the whole protocol stack. One would like to see more parameterization in the medium access layer that would make it possible to modify the contention resolution algorithms from the routing (network) layer and, possibly, from the application.

Chapter 5

Broadcast speedup

5.1 Introduction

The task of broadcasting, as an application-level functionality, is to convey a message from a single source to all other nodes in the network. Broadcasting in ad-hoc wireless networks is a fundamental and frequently used operation. Reactive routing protocols such as DSR [22], AODV [43], TORA [41], depend on broadcasting for route discovery. Proactive protocols, such as DSDV [44] and WRP [36], periodically broadcast updated information about cost metrics. Some protocols even use (selective) broadcasting for the actual forwarding [48].

The simplest approach to broadcasting is *flooding*. The sender broadcasts its message to all its neighbors, which in turn re-broadcast it to their neighbors, and so on, until the packet reaches every node in the network. Needless to say, this simplistic approach is wasteful in bandwidth as well as the scarce battery power because of the large number of superfluous packets that are unnecessarily transmitted. Additionally, in the wireless environment, bursts of traffic resulting from massive broadcasts cause congestion resulting in collisions and even more retransmissions.

Not surprisingly, the problem of efficient broadcasting in the wireless ad-hoc environment has received considerable attention. More efficient algorithms have been proposed, whose main goal is common: minimizing the number of retransmissions while ensuring that the broadcast message reaches all nodes in the network. However, little efforts were made to explicitly address the issue of reducing broadcast latency defined as the time elapsing from the moment a node initiates a broadcast until the last node in the network has received a copy of the message. This performance measure is hardly irrelevant. For example, in the case of a reactive routing protocol, the broadcast latency effectively determines how quickly routes are discovered, which is the primary quality criterion of the underlying routing scheme. This is especially true under conditions of non-trivial mobility, which triggers frequent route changes and, consequently, frequent invocations of the route discovery mechanism. For a proactive protocol, which disseminates the cost metric information via broadcasting, the high cost of this operation will tend to reduce the optimum update frequency and render small updates too costly to disseminate. Consequently, the network will tend to operate with inaccurate

routing information, at a performance penalty that, under non-trivial mobility scenarios, may turn out to be unacceptable.

At first sight, one might naively assume that the objective of minimizing the number of retransmissions in a broadcast scheme automatically implies minimizing the latency: after all, we minimize the total time spent by all nodes on transmitting packets, so what else is there to gain? As we demonstrate in this chapter, this is not true. To conclude that a message should not be retransmitted, a node must (implicitly or explicitly) acquire some information from other nodes. This involves time and, in fact, puts the two objectives at the opposite ends of a trade off game.

The primary contribution of our work is a modification to the backoff component of the IEEE 802.11 family of collision avoidance schemes that effectively reduces the broadcast latency while keeping the number of retransmissions small. Our approach is somewhat reminiscent of the previous work [12] on Quality of Service issues related to fairness and priority scheduling. In our own previous work [47, 48], we proposed two other modifications to IEEE 802.11 to facilitate topology control and increase the reliability of broadcast-based forwarding.

5.2 Broadcast optimization framework

5.2.1 The generic scheme

Practically all broadcast algorithms trying to reduce the number of retransmissions with respect to unconstrained flooding incur delays in the network layer—to let the receiving node monitor the channel before deciding whether the received packet should be retransmitted or dropped. Thus, all such algorithms fit the generic scheme shown in Figure 5.1, which we shall refer to as GBA. The parts in bold, i.e., the setting of the defer timer and the pruning criteria are the specific features of any actual algorithm. In the next section, we describe how those features have been implemented in the existing algorithms for broadcast optimization.

1. *Packet reception: if the packet was seen before, drop it (duplicate discard); otherwise, push it into a queue.*
2. **Set the defer timer** *and wait.*
3. *While waiting, keep receiving broadcast packets from neighbors. When the timer goes off, proceed to 4.*
4. *End of waiting:* ***evaluate pruning criteria.*** *If the packet appears to be redundant, drop it; otherwise, pass it to the MAC layer for retransmission.*

Figure 5.1: GBA: the Generic Broadcast Algorithm

5.2.2 Setting the defer timer

The objective of the defer timer is to provide the node with a window of opportunity to acquire some input to the pruning criteria—by monitoring traffic in the neighborhood. The following three options are typically considered:

1. Constant defer time. This is the simplest approach. Every node sets the defer timer to the same constant time D_c.

2. Random defer time, e.g., [39]. The defer timer is set to a random interval (e.g., uniformly distributed) between 0 and D_{max}.

3. Distance-based defer time, e.g., [58]. The defer timer is set to a value inversely proportional to a power of distance from the sender.

To illustrate the last case, suppose a node u has received a (non-duplicate) broadcast packet from node v. The distance between u and v is d. The communication range is fixed and denoted by R. Then the defer time is calculated as $D = D_{max} \times (R^2 - d^2)/R^2$.

5.2.3 The pruning criteria

The following four pruning criteria were proposed in [39]:

1. Probabilistic approach. The packet is deemed redundant at random. It will be rebroadcast with probability P and ignored with probability $(1 - P)$. In particular, if $P = 1$, the scheme turns into flooding.

2. Counter-based approach. The packet is considered redundant if during the waiting period D, more than N of its copies have been received from the neighbors, where N is a predetermined constant.

3. Distance-based approach: the packet is considered redundant if during the waiting period D, the node receives another copy of the packet transmitted by a neighbor located not further than some predetermined distance d from the node.

4. Location-based approach. Suppose that within the waiting period D, the node has received a number of copies of the packet from some neighbors. The packet is considered redundant if the node is located within the convex hull formed by those neighbors.

The scheme proposed by Lim and Kim [32] boils down to the following pruning criterion:

5. Self-pruning approach (one-hop neighbor list is passed in packet headers). The packet is considered redundant by a node if the list of nodes extracted from the headers of the received packet, covers all the neighbors of the node.

The algorithm introduced by Peng and Wu [42] is based on this criterion:

6. Scalable Broadcast Algorithm (SBA—assuming two-hop neighbor information). Similar to 5, the packet is considered redundant if, based on the combined list of two-hop neighbors extracted from the packets received within the interval D, the node can tell that its retransmission will not reach any new neighbors.

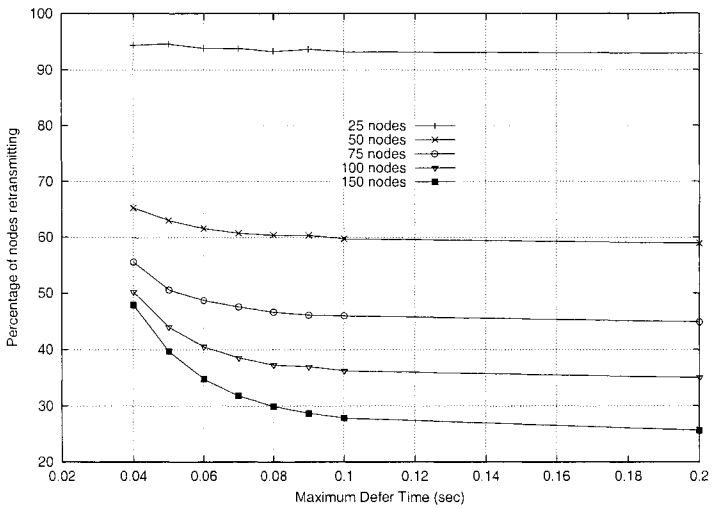
For all the above-listed criteria, except for 1 (the most naive one), the value of D plays a significant role as the base for collecting the relevant information constituting the input to the respective criterion. It seems natural to expect that the longer the interval D, the more information is likely to reach the node before it is forced to make the decision. Note that in all cases it makes sense to randomize D. Otherwise, one-hop neighbors receiving the same packet at the same time will never exchange any feedback among themselves. Also note that criteria 1–4 do not guarantee reachability of all nodes. Their nature is heuristic in the sense that the broadcast is only statistically successful.

5.2.4 The trade off

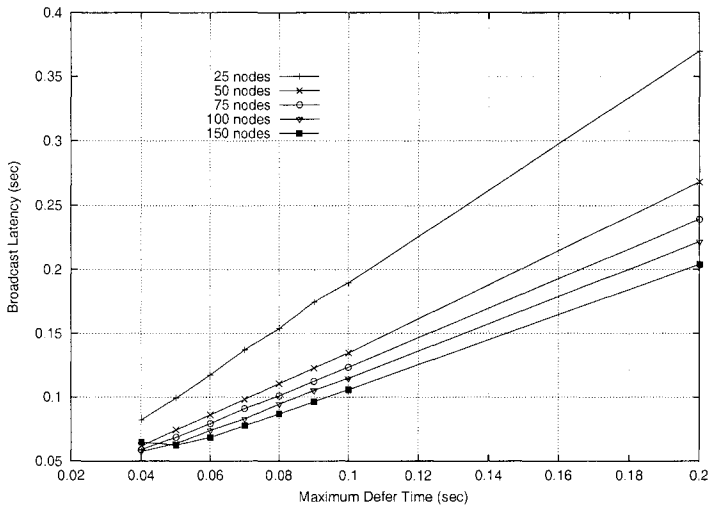
As we mentioned in Section 5.1, our objectives are two-fold: to minimize the number of retransmissions needed to reach all nodes, as well as the latency of the broadcast operation. Although reachability can be guaranteed by criterion 5 [32] or 6 [42], one can easily see that the two objectives contradict one another. This is because increasing the defer interval D allows a node to collect more duplicate packets and thus, possibly, make a better decision. In other words, the longer a node waits before it retransmits a broadcast packet, the higher the likelihood that the retransmission will not take place. Thus, longer waiting time (which increases the latency) will tend to result in fewer retransmissions.

To see how the trade off between the two objectives looks in practice, we ran simulation experiments with networks consisting of various populations of nodes (from 25 to 150) spread over a square region of $670m \times 670m$. Different nodes generated broadcast packets in turns, but, for clarity, no two different broadcast packets existed in the network at the same time. The broadcast algorithm can be succinctly described as <1,2>, where the first value refers to the way of setting D, and the second index identifies the pruning criterion. A node receiving a broadcast packet would defer its retransmission decision by D chosen randomly from $[0, D_{max}]$. Having received more than N packets during that time, the node would refrain from retransmitting the packet. In all our experiments, N was set at 6, while D_{max} changed from 0.04 to 0.2 seconds.

Figure 5.2(a) shows the observed percentage of nodes that retransmitted the received broadcast packets. As the maximum defer time, D_{max}, increases this percentage consistently tends to drop, not surprisingly, the trend being more pronounced for networks consisting of more nodes. On the other hand, as demonstrated by Figure 5.2(b), the broadcast latency increases along with increasing D_{max}.



(a)



(b)

Figure 5.2: Performance of counter-based pruning as a function of maximum defer time w.r.t.: (a) the number of retransmissions, (b) the latency.

5.2.5 The order is important

As noticed in [58], the order in which nodes retransmit a broadcast packet is quite important from the viewpoint of reducing the number of retransmissions. Suppose a node u transmits a broadcast packet. It will be received by all neighbors of u denoted by $\mathcal{N}(u)$. Each node $v \in \mathcal{N}(u)$ will schedule its retransmission after some delay D_u. The configuration of those delays determines the order in which, assuming the respective packets are not deemed redundant, the nodes will retransmit them.

Note that not all of those retransmissions are equally effective and important. For example, if v is located close to u, it can only reach a small additional area compared to what has been already

covered by u. On the other hand, if v is located near the perimeter of u's transmission range, it can relay the packet to more distant regions; thus, its retransmission is likely to cover many additional nodes.

5.2.6 Two components of the defer time

By the above observation, it makes sense to try to give priority to distant nodes, i.e., ones located further from the sender of the broadcast packet. When we look at GBA in Figure 5.1, in isolation from the MAC layer, there appears to be a single place when the node can affect the timing of its retransmission, i.e., step 2, where the node sets up the defer time D. However, in reality, there is another "defer time" contributed to the packet's retransmission delay by the IEEE 802.11 MAC layer [13]. Namely, a node willing to transmit a packet under contention has to wait for a certain number of idle slots chosen at random in the range of $[0, cw-1]$, where cw is the so-called *contention window*. Statistically, different nodes are likely to pick different numbers, which will help them transmit without interference in different time slots. Consequently, the actual delay until the packet is retransmitted can be described as $D_{total} = D + D_{MAC}$, where D_{MAC} is the component incurred by the contention resolution algorithm of the MAC layer.

Only the first component of D_{total}, D, is controllable by GBA, with the second component, D_{MAC}, still being essentially random. Thus, even if the node attempts to follow a sensible priority scheme in setting D, its intentions can still be thwarted by the MAC layer, i.e., the actual retransmission order may be different from the intended one.

One way out could be to always choose $D >> D_{MAC}$, so that the range of the MAC-incurred component is insignificant. This way, however, the proper ordering will be achieved at the cost of increased latency. A better way would be to include the MAC delay in the overall scheme. This will bring about two benefits:

1. The nodes will be able to prioritize their retransmissions without the confusing impact of the MAC layer.

2. The MAC layer will be able to use longer delays without having to worry about impairing the network layer. Thus, it may do a better job as far as avoiding collisions is concerned.

In the next section, we show how to modify the MAC layer of IEEE 802.11 to accomplish this objective.

5.3 MAC-assisted broadcasting

Although the primary motivation for prioritizing retransmission schedules stems from the observation that, generally, distant nodes should transmit first, one can think of a generic priority-based approach to calculating the defer delay D, which may be based on arbitrary (relevant) properties or

phenomena. Our objective at this stage is to incorporate priorities into the calculation of D, which represents the complete defer delay, including the MAC component. In fact, we do not want to separate the two components at all. By crossing the boundary between layers (which has become a standard and extremely useful practice in wireless networking), we can deal with a single delay D elapsing from the moment the node receives a broadcast packet, until it decides to (physically) retransmit it (or ignore).

5.3.1 The outline

The general idea can be stated as follows. Whenever a node receives a broadcast packet, it calculates its priority and immediately passes the packet, along with its priority, to the MAC layer. The network layer reserves the right to revoke the retransmission request if, while the packet is awaiting its turn, the node concludes that the packet is redundant. The MAC layer will push the packet into the interface queue (IFQ) and initiate the backoff procedure. The way the backoff delay is calculated will be biased such that high-priority packets will be delayed for a shorter time than low-priority packets. All the remaining components of the MAC protocol will operate exactly as prescribed in the standard, the only exception being the possibility to cancel the scheduled retransmission upon a signal from the network layer.

5.3.2 The bias mechanism

Suppose that the retransmission priority has been quantized into n discrete levels, e.g., $1, 2, \ldots, n$, where n is a pre-configured parameter, and higher values indicate a higher priority. Assume that the priority of a given packet is i. The backoff delay of the packet is prescribed by the following formula:

$$R = (n - i) \times \frac{cw}{2^{\lceil log_2 n \rceil}} + \lfloor U(0,1) \times \frac{cw}{2^{\lceil log_2 n \rceil}} \rfloor \ ,$$

where $U(0,1)$ denotes uniform distribution between 0 and 1.

Example: If n is a power of 2, the formula is particularly easy to evaluate. Suppose $n = 2$ (only two priority classes) and $cw = 32$. The contention interval is divided into two equal sub-intervals: $[16 - 31]$ and $[0 - 15]$. Note that regardless of the randomization, a packet with priority 1 will be assigned a higher delay than a packet with priority 2.

5.3.3 Distance-based priorities

Let us return to the original motivation behind the prioritized differentiation of the defer delay D, and try to combine it with a reasonable pruning criterion. Let us start with the prescription for calculating priorities. The retransmission priority of a broadcast packet is based on an estimation of the additional area covered by the node contemplating a retransmission.

Suppose that a node v has received a broadcast packet from node u (see Figure 5.3) and is trying to decide whether to retransmit or not. If the distance between u and v is denoted by d,

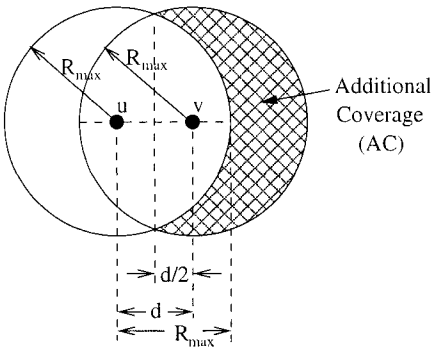


Figure 5.3: Additional coverage area

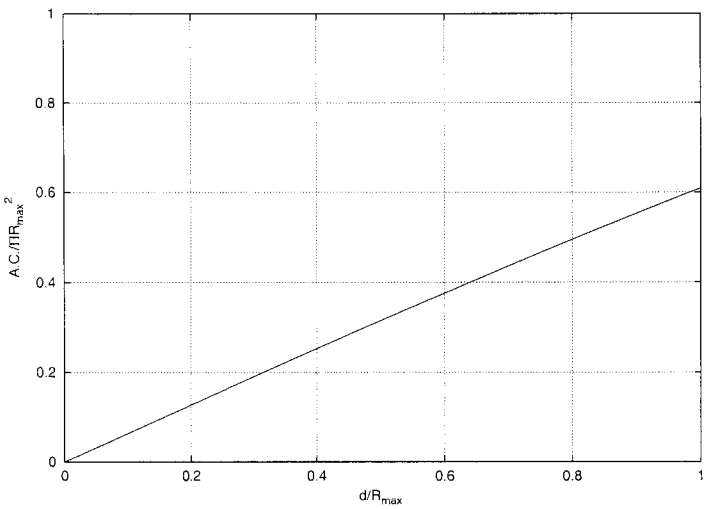


Figure 5.4: AC is an increasing function of distance

and the maximum communication range is R_{max}, then the additional coverage (denoted by AC) is expressed by following equation [39]:

$$\text{AC} = \pi R_{max}^2 - 4\int_{d/2}^{R_{max}} \sqrt{R_{max}^2 - x^2}dx$$

or, alternatively,

$$\text{AC} = 2R_{max}^2 sin^{-1}\frac{d}{2R_{max}} + \frac{d}{2}\sqrt{4R_{max}^2 - d^2} \quad .$$

Figure 5.4 illustrates how the normalized AC, i.e., $\text{AC}/\pi R_{max}^2$, depends on d/R_{max}. It varies (nearly linearly) from 0% to about 61% as d increases from 0 to R_{max}.

In accord with the quantization of priorities, it makes sense to divide the circle representing the coverage of u into n equal-width partitions. Specifically, node v is said to fall into partition i, $1 \leq i \leq n$ iff,

$$\frac{R_{max} \times (i-1)}{n} < d_{sv} \leq \frac{R_{max} \times i}{n} \quad .$$

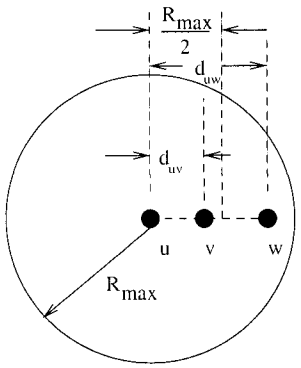


Figure 5.5: A 3-node scenario

Note that each partition corresponds to a circular stripe of width R_{max}/n. Node v will assign priority i to a broadcast packet received from u.

For a useful pruning criterion, we can use directly criterion 3 from Section 5.2.3 [39]. Consider the three nodes shown in Figure 5.5. Both nodes v and w are located within the transmission range of u, i.e., $d_{uv} < R_{max}$ and $d_{uw} < R_{max}$; however, $d_{uv} < R_{max}/2$ while $d_{uw} > R_{max}/2$. Let us assume that the counter threshold $N = 1$, i.e., if the same broadcast packet is received twice, the node will not retransmit the packet. Suppose that u transmits a packet which is received by both v and w. The two nodes calculate their delays D_v and D_w and schedule their retransmissions.

If $D_v < D_w$, then node v will go first. Its transmission will make the counter at w reach 2 before D_w expires, and, consequently, w will not transmit. On the other hand, if $D_v > D_w$, then w will retransmit and v will not.

One can easily see that as long as the priority scheme used in determining D_v and D_w follows the prescription in Section 5.3.2, and $d_{uw} - d_{uv} > R_{max}/n$, we will have the second scenario, in agreement with the principle of preferring nodes with larger additional coverage.

The location-based algorithm described in [39] (see criterion 4 in Section 5.2.3) can also benefit from the distance-based priority scheme. When a node receives the first copy of a broadcast packet, it compares the sender's position to its own location and calculates the additional coverage area of its own retransmission. If that area is less than the prescribed threshold, the packet is ignored immediately. Otherwise, the node sets the retransmission priority according to its distance from the sender and passes the packet to the MAC layer. While waiting for the defer timer to expire, whenever the node receives a new copy of the broadcast packet, it updates its own remaining coverage area. If that area shrinks below the threshold before the queued packet is transmitted, the node will revoke the scheduled transmission.

In this case, the role of the distance-based priority is to speed up the transmission by distant nodes, which are more likely to cover more area and preempt the nodes located closer to the packet source. Even if the unassisted variant of the protocol could eventually find out which nodes should

transmit and which should not, by giving preference to distant nodes, the consensus can be reached quicker, which will have a positive impact on the latency.

5.3.4 Degree-based priorities

Another way of prioritizing retransmissions by adjusting the defer time is to take into account the network density in the node's neighborhood. The general idea is to support retransmissions by nodes with many neighbors. This approach will locally maximize the rate at which broadcast packets spread through the network.

More specifically, a node u receiving a broadcast packet for a possible retransmission will determine the value of the following three parameters: g_u—its own node degree (the number of its neighbors), g_{max}—the maximum node degree among its neighbors (including itself), g_{min}—the minimum node degree among its neighbors (including itself). Let $g_{var} = g_{max} - g_{min} + 1$ be a measure of variation in the node degree in u's neighborhood. The node will set its retransmission priority to i such that:

$$\frac{g_{var} \times (i-1)}{n} + g_{min} \leq g_u < \frac{g_{var} \times i}{n} + g_{min}$$

where $1 \leq i \leq n$ and n, as before, is the assumed number of discrete priority levels. Note that, this formula gives higher priority to nodes having more neighbors.

Degree-based priority schemes, like the one suggested above, will tend to reduce the latency of algorithms in the SBA class [42]. SBA assumes that every node knows the identities of its 2-hop neighbors. Having identified the last sender u of a received broadcast packet, node v can easily determine which of its neighbors are included in the neighbor set of u. As it receives more copies of the packet, v can find out whether those packets have completely covered its own set of neighbors. Should that happen before the defer time expires, v will cancel the scheduled retransmission. It is clear that by giving a head start to the nodes with large sets of neighbors, we will speed up the distribution of the broadcast packet in the neighborhood and faster prune out the redundant retransmissions.

5.4 Experimental results

5.4.1 Performance of biased MAC

To evaluate the performance of our enhancements, we built a detailed simulation model based on *ns-2* [1] with wireless extensions. A modified version of the distributed coordination function (DCF) of the IEEE standard 802.11 [13], was used as the MAC layer. The modifications consisted in adding the capability to generate biased backoff delays, as described in Section 5.3.2.

We deployed networks with $25 - 150$ nodes (with the step of 25) over a flat square area of $670m \times 670m$. The distribution of nodes was uniform over the entire area. The transmission range was the same for all nodes and set as $250m$. Three pruning algorithms were implemented: the

counter-based scheme with the threshold $N = 6$, the location-based scheme using the convex hull mechanism proposed in [39], and SBA.

Note that neither the counter-based nor the location-based schemes guarantee that every broadcast packet will always reach all nodes. The threshold values during our experiments were suitably chosen such that the reachability of the broadcast packets in all scenarios were more than 98%.

Each node in the network generated a broadcast packet in turn, but only one broadcast packet (possibly in multiple copies) was present in the network at any given time. The performance measures were collected for each broadcast and then, at the end of run, the averages of those measures were taken. The points used to make up the curves were the averages of those averages collected over five independent experiments.

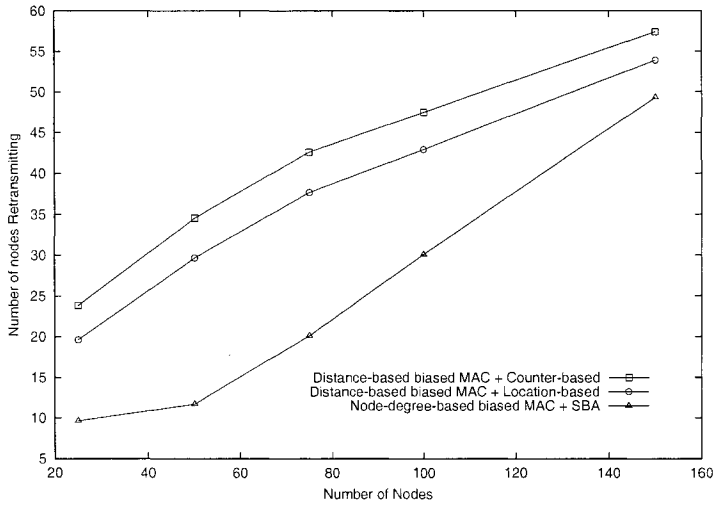
Figure 5.6 illustrates the effect of MAC-biasing on different pruning strategies. Both distance-based and degree-based priority schemes were investigated, with four priority levels in each case.

Figure 5.6(a) shows the average number of nodes participating in retransmissions to complete the broadcast. The combination of the degree-based priority scheme with SBA pruning exhibits the best performance. SBA algorithm uses knowledge of neighborhood which ultimately helps it to offer better results than location based and counter based method. With SBA pruning algorithm, a node rebroadcasts only if all its neighbours have not received a copy of it by some previous rebroadcasts. On the other hand, location-based and counter-based pruning schemes probabilistically predicts the usefulness of a rebroadcast based on the values of some key parameters, i.e., additional coverage area or counter. Therefore, a redundant broadcast may go undetected unlike the SBA algorithm. With the distance-based priority scheme, the location-based pruning shows better performance than the counter-based approach.

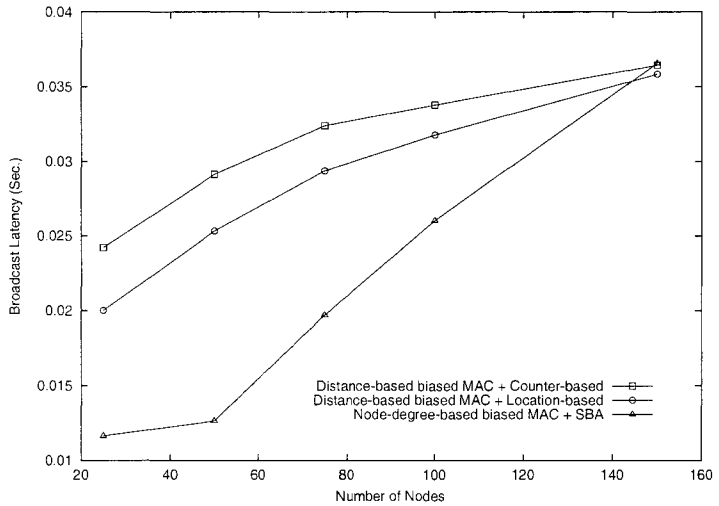
Figure 5.6(b) shows the broadcast latency. Again the combination of the degree-based priority scheme and SBA exhibits the best performance. The location-based pruning also has a lower latency than the counter-based scheme with the same distance-based biased MAC.

According to the IEEE 802.11 standard, a node having a packet to transmit, starts with the contention window (cw) of 32 slots. Owing to the fact that network-layer delays needed by the priority schemes tend to be longer than typical MAC-layer delays (needed solely for contention resolution), it may make sense to start from a larger contention window. From the viewpoint of contention resolution, a larger window can only help. Thus, we ran a series of experiments to study the impact of a large initial contention window on the performance of our modifications.

With cw set to 256, we were able to accommodate more priority classes, so we increased their number from 4 to 8. The results (with SBA used as the pruning scheme) are shown in Figure 5.7. They indicate that the number of retransmissions can be reduced even further, but only at the cost of increased latency.



(a)



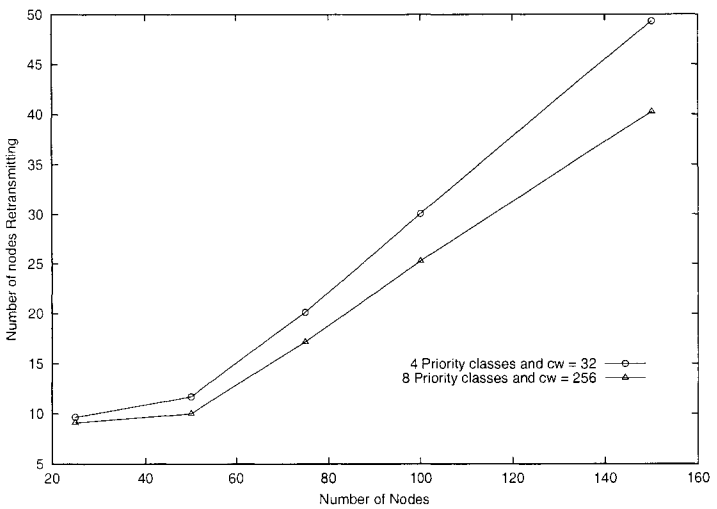
(b)

Figure 5.6: Performance of biased MAC combined with different pruning schemes w.r.t.: (a) the number of retransmissions, (b) the latency.

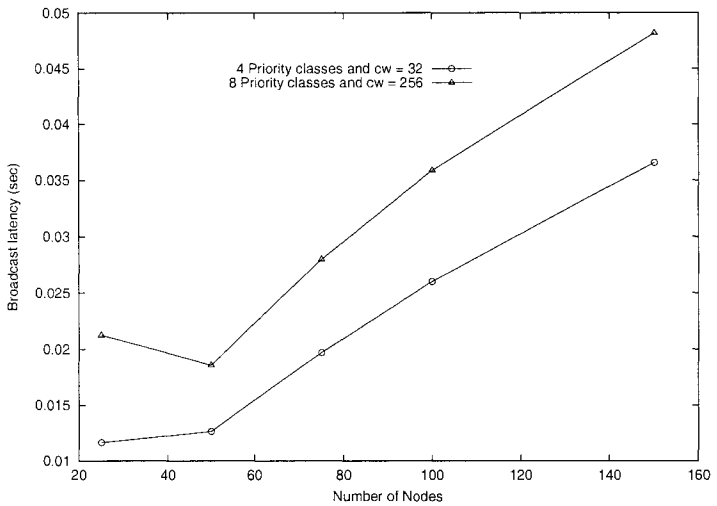
5.4.2 Comparison with other defer schemes

Figures 5.8–5.13 compare the performance of MAC-biasing with other defer schemes: constant, random, and distance-based—in combination with different pruning techniques. The standard 802.11 MAC protocol was used in the last three cases.

The results are quite consistent: our approach offers a significantly lower latency than the other schemes, which, in some cases, comes at the price of a (slightly) increased number of retransmissions. However, the penalty in the number of retransmissions only occurs in those situations when the limit on defer time is high, thus encouraging longer delays during which the nodes have plenty of time to monitor activities in the neighborhood before deciding whether to retransmit. Note that even



(a)



(b)

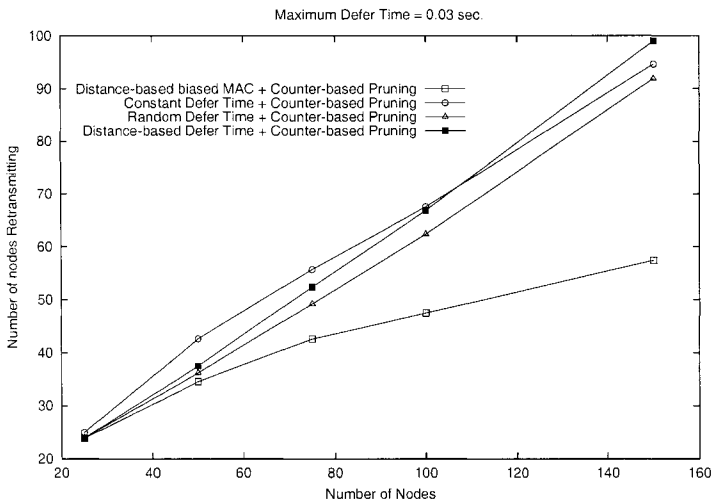
Figure 5.7: Performance of biased MAC with different numbers of priority classes and initial contention window size w.r.t.: (a) the number of retransmissions, (b) the latency.

in the worst case (Figure 5.9(a)), the number of retransmissions is only about 20% worse compared to the best result (for random defer time), while the latency offered by our scheme is lower by the factor of two.

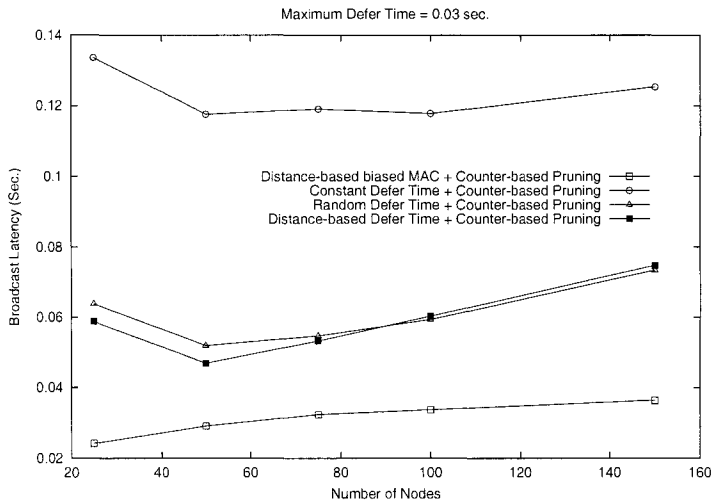
5.5 Chapter summary

We have proposed a simple modification to the backoff algorithm of IEEE 802.11 aimed at improving the performance of broadcasting in wireless networks. Our enhancement considerably reduces the latency of broadcasting while maintaining a low number of retransmissions.

As a future work, we plan to investigate the performance of our scheme used as a supplement



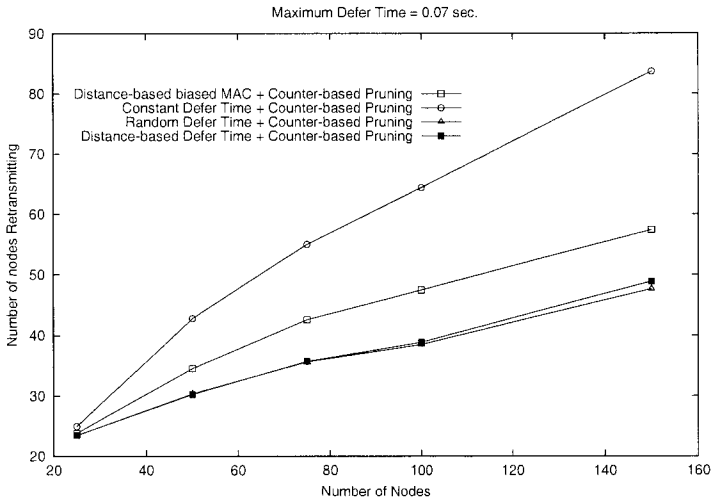
(a)



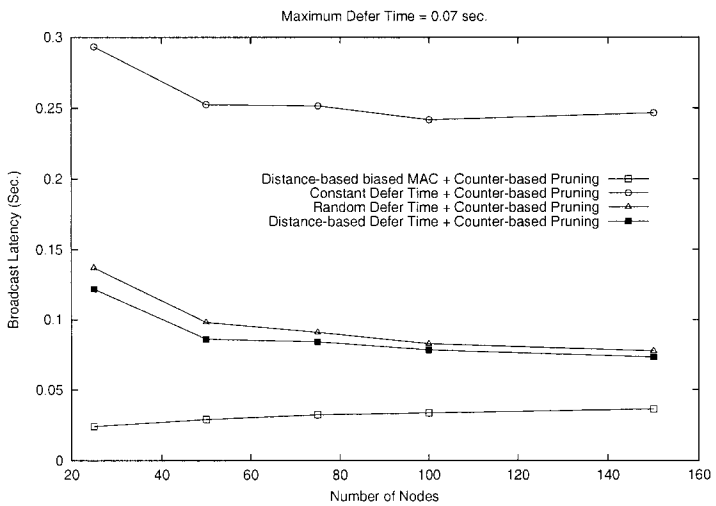
(b)

Figure 5.8: Comparison of different defer schemes combined with counter-based pruning w.r.t: (a) the number of retransmissions, (b) the latency. The maximum allowable defer time is 0.03 sec.

to some well-established reactive routing protocols, such as AODV [45] and DSR [22], to assist them in route discovery. It will be interesting to see how much the improvements to this part of the routing protocol will benefit the performance of the entire system. At the same time, we are modifying DSDV [44] to use our broadcasting technique for propagating routing tables.

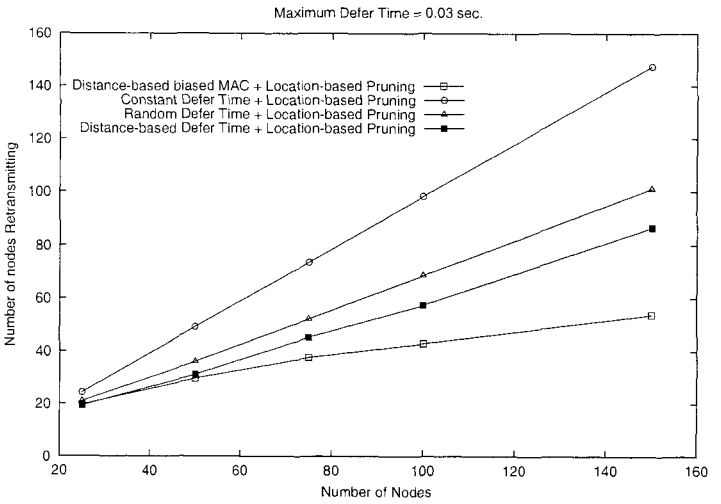


(a)

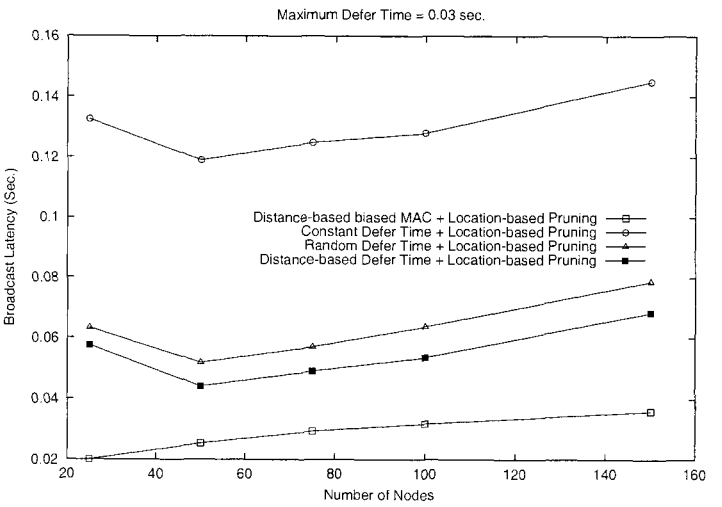


(b)

Figure 5.9: Comparison of different defer schemes combined with counter-based pruning w.r.t.: (a) the number of retransmissions, (b) the latency. The maximum allowable defer time is 0.07 sec.

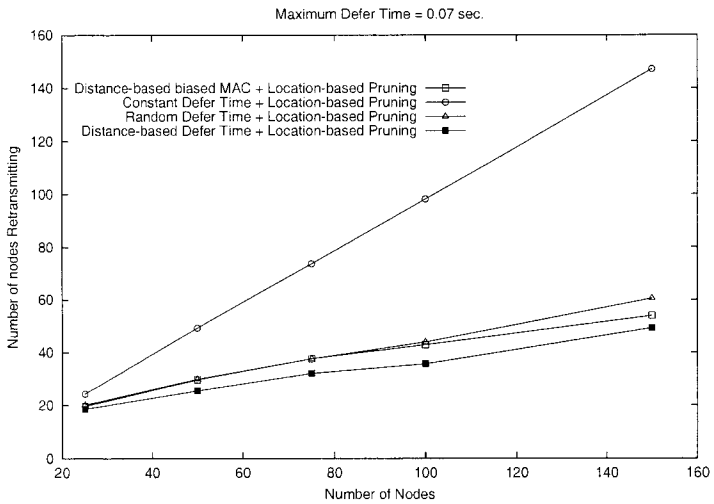


(a)

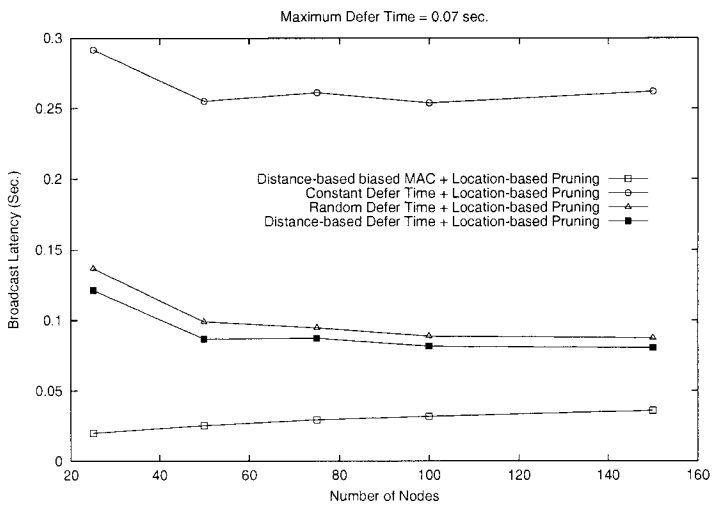


(b)

Figure 5.10: Comparison of different defer schemes combined with location-based pruning w.r.t: (a) the number of retransmitting nodes, (b) broadcast latency. The maximum allowable defer time is 0.03 sec.

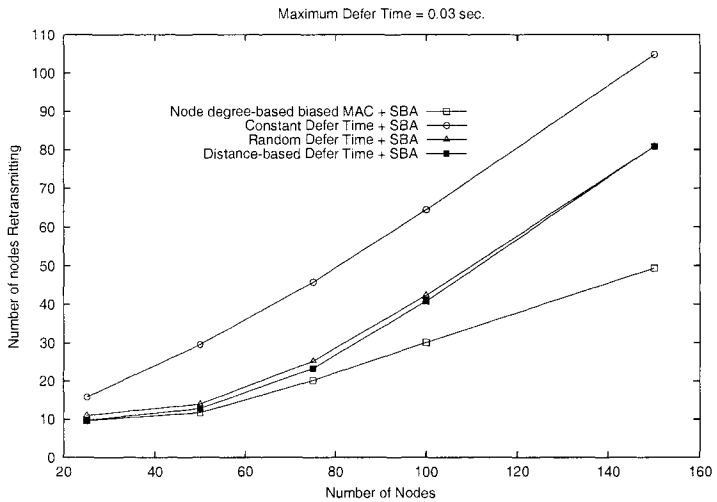


(a)

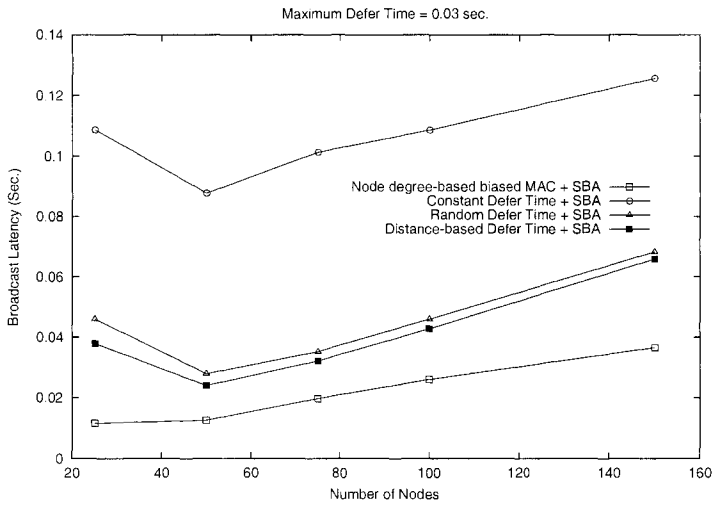


(b)

Figure 5.11: Comparison of different defer schemes combined with location-based pruning w.r.t.: (a) the number of retransmitting nodes, (b) broadcast latency. The maximum allowable defer time is 0.07 sec.

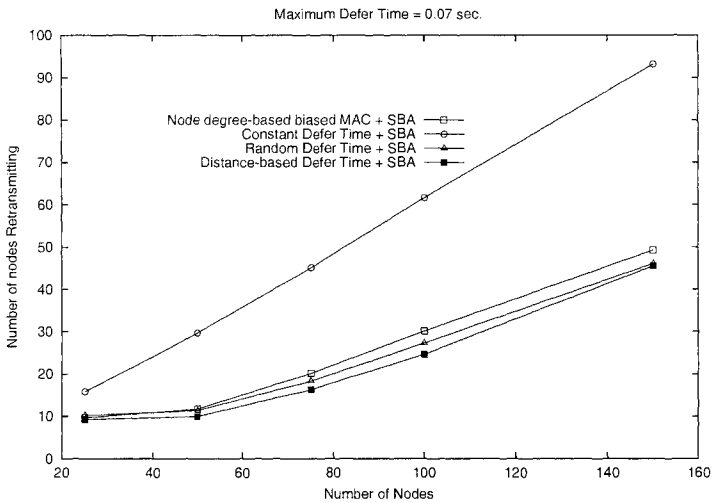


(a)

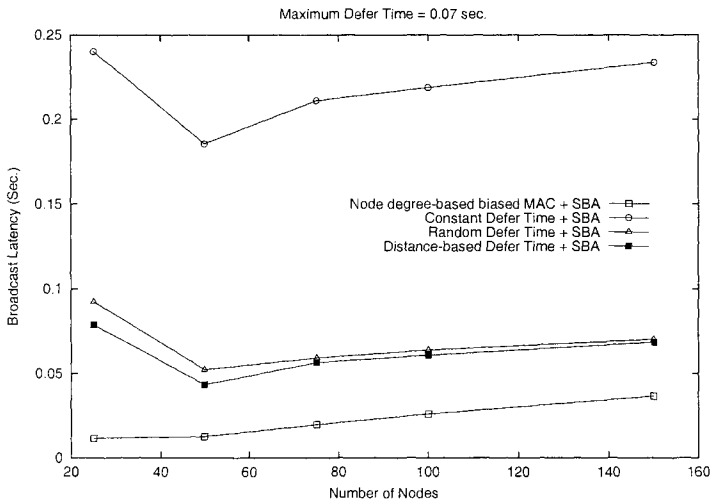


(b)

Figure 5.12: Comparison of different defer schemes combined with SBA w.r.t.: (a) the number of retransmissions, (b) the latency. The maximum allowable defer time is 0.03 sec.



(a)



(b)

Figure 5.13: Comparison of different defer schemes combined with SBA w.r.t.: (a) the number of retransmissions, (b) the latency. The maximum allowable defer time is 0.07 sec.

Chapter 6

Contributions, conclusions and future work

6.1 Overview of results

This dissertation considers several problems regarding wireless ad hoc networks. First, we presented *TARP*, the Tiny Ad-hoc Routing Protocol. This protocol provides routing for applications running on inexpensive ad-hoc stations with controllable memory, processing and power requirements. TARP is based on two simple algorithms whose purpose is to limit the number of packets in the network. The first, the *Duplicate Discard* algorithm, is used to control flooding, while the other, the *Sub-optimal Path Discard* algorithm, is used to drop packets traveling on sub-optimal (longer than the shortest) paths. The way the protocol operates may be easily controlled via modes of operation. These modes may be either set statically by the application designer, or dynamically by the application itself, in reaction to changes in the environment. The simplicity of the algorithms, as well as ease of adjusting their parameters by the application, provide TARP with a great deal of flexibility and make it resilient to problems due to frequent changes of topology—one of the main characteristics of ad-hoc networks. There are no routing tables, no complex methods of maintaining the route once it is established, and no reactions to topology changes. No control messages are exchanged between stations.

Next we pointed out a deficiency of IEEE 802.11 based MAC layer in supporting broadcast-based routing. Broadcast packets are not transmitted as reliably as unicast packets. The absence of feedback from the recipients makes it impossible for the sender of a broadcast packet to take any retransmission decision. We tackled this issue with an innovative idea of 'fuzzy acknowledgment' and some cross layer interaction between the MAC layer and the network layer. Incorporating the 'fuzzy acknowledgment' mechanism into the base protocol TARP, we have shown how to enhance the reliability of broadcast packets in ad-hoc wireless environment.

We worked with topology control aimed at power saving in routing protocols. The topology control algorithm that we developed here builds a platform for the routing protocols so that they can

choose minimum-energy paths on which to route the packets. Our algorithm discovers the optimal topology with less overhead than other existing algorithms.

Finally, we presented a simple modification to the backoff algorithm of IEEE 802.11 aimed at improving the performance of broadcasting in wireless networks. Our enhancement considerably reduces the latency of broadcasting while maintaining a low number of retransmissions.

6.2 Cross layer interaction

Suppose we like to develop a complete ad hoc networking infrastructure combining the routing scheme, the MAC layer enhancement, the power control algorithm and the scheme for faster broadcast altogether that we have proposed in this dissertation. While combining, definitely we need to accommodate several cross layer inter-actions at the same time. For example, the power control algorithm requires that the back-off mechanism of MAC layer should be done in such a way so that closer nodes get priority over distant nodes. On the other hand, the scheme for faster broadcast requires the opposite order, i.e., the distant nodes should get priority over closer nodes. We can accommodate these two contradictory goals by parameterizing the MAC layer's back-off functions with the help of one more parameter. The value of this parameter will determine the order of biasing and the upper layers will set its value to achieve desired ordering.

For the routing protocol, we have proposed a "fuzzy acknowledging mechanism" to enhance the reliability of broadcast packets. If we also like to accommodate unreliable broadcasts for some other purposes then we can support both with the help of another parameter. The value of this parameter set by the upper layers will tell the MAC layer whether to broadcast with "fuzzy acknowledging mechanism" or without.

Thus, through cross-layer interactions between layers, we may facilitate the routing, the enhanced MAC reliability, the power control for energy efficiency and the faster broadcast on top of the same MAC layer.

6.3 Future work

In the future, the routing protocol can be extended to support multicasting. With the current implementation, broadcasting can be easily handled because the protocol uses controlled flooding as an underlying principle. But for now it is assumed that higher layers will be responsible for multicasting. The protocol tends to suffer when we increase traffic load. Adapting the protocol to handle highly intensive traffic flows is another matter for further investigation. Another problem with TARP is, with a higher number of nodes, the convergence of routing in the direction of the shortest path seems to be slow. Mechanisms for faster convergence can be further investigated.

The power control algorithm presented in this thesis preserves all minimum-energy paths between all pairs of nodes, and assumes that any routing protocol running on top of it will route

packets over those paths. But the performance of the power control algorithm was not evaluated in conjunction with any routing protocols. As future work, we are planning to measure the performance of a modified DSDV protocol in conjunction with our power control algorithm to determine the power savings that could be made using our algorithm. The modified version of DSDV will use power requirements as cost metric instead of hop counts. Another obvious idea would be to use the power control algorithm with the routing protocol that we presented in this thesis. Combining the power control algorithm and TARP may give rise to some interesting research issues. The reduction of the power level by each node ultimately reduces the average node degree, interference and contention which are particularly suitable for a TARP-like flooding-based protocol. On the other hand, as the size of each node's neighborset is reduced by reducing its power level, the source node now sees fewer paths to the destination. Fewer path options may reduce the amount of successfully delivered packets. It would be interesting to further investigate whether there exists any trade-off between power saving and the number of packets successfully delivered in TARP.

It would be also interesting to evaluate our proposed power control algorithm in a mobile environment. Given that the network is static, the minimum-energy path-preserving graph has to be constructed only once, at the time of network formation. But, in the mobile environment, the same algorithm would run periodically to accommodate the changes in a node's position. In highly mobile environment, location updates would be more frequent. As a result, every node in such an environment should run the algorithm more frequently to calculate their correct neighborset. It is not only mobility that makes the reconstruction of the low overhead graph necessary. In a static scenario, interference and battery status can also require this reconstruction. However, the impact of these factors on the performance of the presented technique was not evaluated. Addressing these issues in the future would add value to this research effort.

Evaluating the power control algorithm on a testbed is another possible direction of further research. There are some hardware limitations of Network Interface Cards. For example, the CISCO-Aironet 350 series cards, impose a huge latency for power change [26]. The limitation is not rooted in the underlying electronics, rather in the way the firmware has been programmed. In a static environment, the optimal power level is set only once—after running the algorithm to determine appropriate power level. But in mobile environment, as the neighborsets are continuously changing, the power levels of each node need to be set more than once. Frequent changes in a node's position will cause frequent changes in its power level. As a result, the latency of changing the power level in certain hardware may degrade the performance of the algorithm in mobile environment.

We also plan to investigate the performance of our broadcasting scheme used as a supplement to some well-established reactive routing protocols, such as AODV [45] and DSR [22] to assist them in route discovery. It would be interesting to see how the improvements to this part of the routing protocol benefit the performance of the entire system.

Bibliography

[1] The Network Simulator: NS-2: notes and documentation. http://www.isi.edu/nsnam/ns/.

[2] N. Abramson. The ALOHA system—another alternative for computer communications. In *Fall Joint Computer Conference, AFIPS Conference Proceedings*, volume 37, pages 281–285, 1970.

[3] E. Akhmetshina, P. Gburzynski, and F. Vizeacoumar. PicOS: A tiny operating system for extremely small embedded platforms. In *Proceedings of ESA'03*, pages 116–122, Las Vegas, jun 2003.

[4] G. Bianchi. Performance analysis of the IEEE 802.11 distributed coordination function. *IEEE Journal on Selected Areas in Communications*, 18(3):535–547, 2000.

[5] P. Bose, P. Morin, I. Stojmenovic, and J. Urrutia. Routing with guaranteed delivery in ad hoc wireless network. *ACM/Kluwer Wireless Networks*, 7(6), 2001.

[6] D.A. Broch, J.and Maltz, D.B. Johnson, Y-C Hu, and J. Jetcheva. A performance comparison of multi-hop wireless ad-hoc network routing protocols. In *Proceedings of MOBICOM'98*, pages 85–97, October 1998.

[7] A. Chandra, V. Gummalla, and J. O. Limb. Wireless medium access control protocols. *IEEE Communications Surveys*, Second Quarter 2000.

[8] C.C. Chiang. Routing in clustered multihop, mobile wireless networks with fading channel. In *Proceedings of the IEEE SECON*, pages 197–211, April 1997.

[9] I. Chlamtac, A.D. Myers, V.R. Syrotiuk, and G. Zaruba. An adaptive media access control (MAC) for reliable broadcast in wireless networks. In *Proceedings of IEEE International Conference on Communications (ICC '00)*, June 2000.

[10] B. P. Crow, I. Wadjaja, J. G. Kim, and P. T. Sakai. IEEE 802.11 wireless local area networks. *IEEE Communications Magazine*, pages 116–126, September 1997.

[11] B. Das, R. Sivakumar, and V. Bharghavan. Routing in ad-hoc networks using a virtual backbone. In *Proc. of International conference on Computer Communications and Networks*, pages 1–20, Sept. 1997.

[12] J. Deng and R.-S. Chang. A priority scheme for IEEE 802.11 DCF access method. *IEICE Trans. Commun.*, E82-B(1), January 1999.

[13] IEEE Standards Department. Wireless LAN medium access control (MAC) and physical layer (PHY) specifications, 1997. IEEE standard 802.11-1997.

[14] D. Eckhardt and P. Steenkiste. Measurement and analysis of the error characteristics of an in-building wireless network. In *Proceedings of the ACM SIGCOMM'96 Conference*, pages 243–254, October 1996.

[15] K.R. Gabriel and R.R. Sokal. A new statistical approach to geographic variation analysis. *Systematic Zoology*, 18:259–278, 1969.

[16] M. Gunes, U. Sorges, and I. Bouazizi. ARA—the ant-colony based routing algorithm for MANETs. In *Proceedings of International Workshop on Ad-hoc Networking (IWAHN)*, Vancouver, British Columbia, Canada, August 2002.

[17] Z. J. Haas. A new routing protocol for the reconfigurable wireless networks. In *Proceedings of ICUPC'97*, San Diego, CA, October 1997.

[18] H. Hassanein and A. Zhou. Routing with load balancing in wireless ad hoc networks. In *Proceedings of ACM MSWiM*, Rome, Italy, July 2001.

[19] Z. Huang, C. Shen, C. Srisathapornphat, and C. Jaikaeo. Topology control for ad hoc networks with directional antennas. In *Proc. IEEE Int. Conference on Computer Communications and Networks*, pages 16–21, 2002.

[20] F. Ingelrest, D. Simplot-Ryl, and I. Stojmenovic. Energy-efficient broadcasting in wireless mobile ad hoc networks. *Resource Management in Wireless Networking*, pages 543–582, 2005.

[21] H. Li J. Wu. A dominating-set-based routing scheme in ad hoc wireless networks. *Telecommunication Systems*, 18(1):13–36, 2001.

[22] D. B. Johnson and D. A. Maltz. Dynamic Source Routing in ad hoc wireless networks. In Imielinski and Korth, editors, *Mobile Computing*, volume 353. Kluwer Academic Publishers, 1996.

[23] P. Karn. MACA—a new channel access protocol for packet radio. In *In Proc. of ARRL/CRRL Amateur Radio 9th Computer Networking Conference*, September 1990.

[24] B. Karp and H. Kung. GPSR: Greedy Perimeter Stateless Routing for wireless networks. In *ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom)*, Boston, USA, August 2000.

[25] V. Kawadia and P. R. Kumar. Power control and clustering in ad hoc networks. In *Proceedings of INFOCOM 2003*, San Francisco, USA, April 2003.

[26] Vikas Kawadia. *Protocols and architecture for wireless ad-hoc networks*. PhD thesis, University of Illinois at Urbana-Champaign, 2004.

[27] S.-J. Lee and Gerla M. Dynamic load-aware routing in ad hoc networks. In *Proceedings of IEEE ICC'01*, Helsinki, Finland, June 2001.

[28] L. Li and J. Halpern. Minimum energy mobile wireless networks revisited. In *Proceedings of IEEE Conference on Communications (ICC '01)*, 2001.

[29] L. Li, J. Y. Hlapern, P. Bahl, Y. Wang, and R. Watenhofer. Analysis of a con-based topology control algorithm for wireless multi-hop networks. In *In ACM Symposium on Principle of Distributed Computing (PODC)*, 2001.

[30] N. Li, J. C. Hou, and L. Sha. Design and analysis of an MST-based topology control algorithm. In *Proceedings of INFOCOM*, 2003.

[31] D. Lichtenstein. Planar formulae and their uses. *SIAM Journal on Computing*, 11(2):329–343, 1982.

[32] H. Lim and C. Kim. Multicast tree construction and flooding in wireless ad hoc networks. In *ACM International Workshop on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWIM)*, 2000.

[33] W. Lou and J. Wu. On reducing broadcast redundancy in ad -hoc wireless networks. *IEEE Transactions on Mobile Computing*, 1(2):111–123, 2002.

[34] M. Mauve, J. Widmer, and Hartenstein H. A survey on position-based routing in mobile ad-hoc networks. *IEEE Network Magazine*, 15(6):30–39, 2001.

[35] J. P. Monks, V. Bhargavan, and W. M. Hwu. A power controlled MAC protocol for wireless packet networks. In *Proceedings of INFOCOM*, pages 219–228, 2001.

[36] S. Murthy and J. J. Garcia-Luna-Aceves. An efficient routing protocol for wireless networks. *ACM Mobile Networks and Applications Journal*, pages 183–197, October 1996.

[37] A. D. Myers, G. V. Zruba, and V. R. Syrotiuk. An adaptive generalized transmission protocol for ad hoc networks. *MONET*, 7(6):439–502, 2002.

[38] S. Narayanaswamy, V. Kawadia, R. S. Sreenivas, and P. R. Kumar. Power control in ad-hoc networks: Theory, architecture, algorithm and implementation of the COMPOW protocol. In *Proceedings of the European Wireless Conference - Next Generation Wireless Networks:*

Technologies, Protocols, Services and Applications, pages 156–162, Florence, Italy, February 2002.

[39] S.-Y. Ni, Y.-C. Tseng, Y.-S. C. Hen, and J.-P. Sheu. The broadcast storm problem in a mobile ad hoc network. In *Mobicom*, 1999.

[40] W. Olesinski, A. Rahman, and P. Gburzynski. TARP: a Tiny Ad-hoc Routing Protocol for wireless networks. In *Australian Telecommunication, Networks and Applications Conference (ATNAC)*, Melbourne, Australia, December 2003.

[41] V.D. Park and M.S. Cors, on. A performance comparison of TORA and ideal link state routing. In *Proceedings of IEEE Symposium on Computers and Communications '98*, June 1998.

[42] W. Peng and X. Lu. On the reduction of broadcast redundancy in mobile ad hoc networks. In *Mobihoc*, 2000.

[43] C. Perkins, E. Belding Royer, and S. Das. Ad-hoc On-demand Distance Vector Routing (AODV), February 2003. Internet Draft: draft-ietf-manet-aodv-13.txt.

[44] C.E. Perkins and P. Bhagwat. Highly dynamic Destination-Sequenced Distance Vector routing (DSDV) for mobile computers. In *Proceedings of SIGCOMM'94*, pages 234–244, August 1993.

[45] C.E. Perkins and E.M. Royer. Ad-hoc On-demand Distance Vector Routing (AODV). In *Proceedings of the IEEE workshop on Mobile Computing Systems and Applications (WMCSA)*, pages 90–100, 1999.

[46] A. Qayyum, L. Viennot, and A. Laouiti. Multipoint relaying: An efficient technique for flooding in mobile wireless networks, 2000. Technical Report 3898, INRIA–Rapport de recherche.

[47] A. Rahman and P. Gburzynski. MAC-assisted topology control for ad-hoc wireless network. *International Journal of Communication Systems*, 2005.

[48] A. Rahman, W. Olesinski, and P. Gburzynski. Controlled flooding in wireless ad-hoc networks. In *Proceedings of IWWAN'04*, Oulu, Finland, June 2004.

[49] Rajmohan Rajaraman. Topology control and routing in ad hoc networks: A survey. In *Proceedings of SIGACT News*, June 2002.

[50] R. Ramanathan and R. Rosales-Hain. Topology control of multihop wireless networks using transmit power adjustment. In *In Proc. of IEEE INFOCOM 2000*, pages 404–413, Tel Aviv, Israel, March 2000.

[51] T.S. Rappaport. Wireless communications: principles and practice, 1996. Prentice Hall.

[52] V. Rodoplu and T. Meng. Minimum energy mobile wireless networks. *IEEE Journal of Selected Areas in Communications*, 17(8):1333–1344, 1999.

[53] S. Singh and C.S. Raghavendra. Power efficient MAC protocol for multihop radio networks. In *Proceedings of PIMRC*, pages 153–157, 1998.

[54] S. Singh, M. Woo, and C.S. Raghavendra. Power-aware routing in mobile ad hoc networks. In *Proceedings of MobiCom*, pages 181–190, 1998.

[55] R. Sisodia, B. Manoj, and C. Murthy. A preferred link based routing protocol for wireless ad hoc networks. *IEEE/KICS Journal of Communication Networks*, 4(1):14–21, 2002.

[56] I. Stojmenovic, M. Seddigh, and J. Zunic. Dominating sets and neighbour elimination based broadcasting algorithms in wireless networks. *IEEE Transactions on Parallel and Distributed Systems*, 13(1):14–25, January 2002.

[57] I. Stojmenovic and J. Wu. Broadcasting and activity scheduling in ad hoc networks. *Mobile Ad Hoc Networking*, pages 205–229, 2004.

[58] M. T. Sun, W. C. Feng, , and T. H. Lai. Location aided broadcast in wireless ad hoc networks. In *Proceedings of GLOBECOM'01*, 2001.

[59] Z. Tang and Garcia-Luna-Aceves. A protocol for topology-dependent transmission scheduling in wireless networks. In *Proceedings of the IEEE WCNC*, pages 1333–1337, September 1999.

[60] F. Tobagi and L. Kleinrock. Packet switching in radio channels, carrier sense multiple access models and their throughput delay characteristics. *IEEE Transactions on Communications*, 23(12):1400–1416, December 1975.

[61] C-K Toh. A novel distributed routing protocol to support ad-hoc mobile computing. In *Proceedings of IEEE 15th Annual International Phoenix Conf. on Comp. and Comm.*, pages 480–486, March 1996.

[62] B. Tuch. Development of WaveLAN, an ISM band wireless LAN. *AT&T Technical Journal*, 72(4):27–33, July/Aug 1973.

[63] R. Wattenhofer, L. Li, P. Bahl, and Y.-M. Wang. Distributed topology control for wireless multihop ad-hoc networks. In *INFOCOM*, pages 1388–1397, 2001.

[64] J. Weinmiller, M. Schlager, A. Festag, and A. Wolisz. Performance study of access control in wireless LANs IEEE 802.11 DFWMAC and ETSI RES 10 HIPERLAN. *Mobile Networks and Applications*, 2:55–67, 1997.

[65] B. Williams and T. Camp. Comparison of broadcasting techniques for mobile ad hoc networks. In *MOBIHOC*, 2002.

[66] J. Wu and F. Dai. Broadcasting in ad hoc networks based on self-pruning. In *Proc. of INFOCOM*, March 2003.

[67] J. Wu and W. Lou. Forward-node-set-based broadcast in clustered mobile ad hoc networks. *Wireless Communications and Mobile Computing*, 3(2):155–173, 2003.

[68] K. Wu and J. Harms. Load-sensitive routing for mobile ad hoc networks. In *Proceedings of IEEE ICCCN*, Arizona, USA, October 2001.

[69] A. Yao. On constructing minimum spanning trees in k-dimensional spaces and related problems. *SIAM Journal on Computing*, pages 721–736, 1982.

[70] C. Yu, B. Lee, and H. Y. Youn. Energy efficient routing protocols for mobile ad hoc networks. *Wireless Communications and Mobile Computing*, 3(8):959–973, 2003.