INTERNATIONAL JOURNAL OF ADVANCE RESEARCH, IDEAS AND INNOVATIONS IN TECHNOLOGY

ISSN: 2454-132X
Impact factor: 4.295
(Volume 4, Issue 4)
Available online at: www.ijariit.com

Medium Access Control and Quality-of-Service Routing for Mobile Ad Hoc Networks

Computer Science and Engineering

ABSTRACT

Title of Dissertation: Medium Access Control and Quality-of-Service Routing for Mobile Ad Hoc Networks

A mobile ad hoc network is an autonomous system consisting solely of mo-bile terminals connected with wireless links. This type of network has received considerable interest in recent years due to its capability to be deployed quickly without any fixed infrastructure. Nodes self-organize and re-con gure as they join, move, or leave the network. How to design distributed protocols capable of handling the dynamic nature of these networks is an interesting but di cult topic.When TDMA is used, distributed protocols are needed to generate transmis-sion schedules. An important issue is how to produce a schedule quickly. This is critical when the network is large or the network changes frequently. Here we develop two fully distributed protocols for generating or updating TDMA schedules. Contention is incorporated into the scheduling protocols for them to work independently of the network size. The schedule can be generated at multiple parts of the network simultaneously. In the Five-Phase Reserva-tion Protocol (FPRP), a broadcast schedule is produced when nodes contend among themselves using a new ve-phase message exchange mechanism. In the Evolutionary-TDMA scheduling protocol (E-TDMA), schedules are updated when nodes contend to reserve transmission slots of di erent types (unicast, multicast, broadcast). Both are scalable protocols suitable for large or dynamic networks. Another issue related to medium access control is transmission power control. Our contribution to power control is to develop a channel probing scheme for networks applying power control, which allows a node to probe a channel and estimate the channel condition. It can be used for dynamic channel allocation in a TDMA or FDMA system, or admission control in a DS/CDMA system. It is a fully distributed scheme which requires little communication overhead. Multiple links can probe a channel simultaneously and each makes individual yet correct decisions.The last topic is Quality-of-Service routing. An e cient distributed scheme is developed to calculate the end-to-end bandwidth of a route. By incorporating this scheme with the AODV protocol, we developed an on-demand QoS routing protocol which can support CBR sessions by establishing QoS routes with re-served bandwidth. It repairs a route when it breaks.

Load balancing and route redundancy are also achieved. It is applicable for small networks or short routes under relatively low mobility.

Table of Contents

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

1.1 Background

Wireless networks have experienced unprecedented development in the past decade. One of the most rapidly developing areas is mobile ad hoc networks (also called mobile packet radio networks or mobile multihop wireless networks). Physically, a mobile ad hoc network consists of a number of geographically-distributed, potentially mobile nodes sharing a common radio channel. Com-pared with other type of networks, such as cellular networks or satellite networks, the most distinctive feature of mobile ad hoc networks is the lack of any fixed infrastructure. The network is consisted of mobile nodes only, and a network is created \on the fly" as the nodes transmit with each other. The network does not depend on a particular node and dynamically adjusts as some nodes join or others leave the network. As a consequence, a network like this is both flexible and robust. In a hostile environment where a fixed communication infrastructure is unreliable or unavailable, such as in a battle eld or in a natural disaster area struck by earthquake or hurricane, an ad hoc network can be quickly deployed and provide limited but much needed communications. As a matter of fact, the concept of mobile ad hoc networks is not new. It dates back to the DARPA packet radio network program in the 1970's [1, 2, 3]. The renewed interest in these networks in recent years is largely due to the development of mobile com-puting and wireless technology. While the military is still a major driving force behind the development of these networks, ad hoc networks are quickly nding new applications in civilian areas. Ad hoc networks will enable people to ex-change data in the eld or in a class room without using any network structure except the one they create by simply turning on their computers or PDAs, or enable a flock of robots (UAVs, satellites, etc.) to form a self-organizing group and collectively perform some task. A large, mesh-shaped network can replace or enhance a cellular network. RoofTop Networks [4], which forms a multihop wireless network by installing radios at roof top and provides residential wire-less internet services, and Bluetooth Technology [5], which replaces cables with wireless networking, can both be viewed as special kinds of these networks. As wireless communication increasingly permeates everyday life, new applications for mobile ad

hoc networks will continue to emerge and become an important part of the communication structure.As mobile ad hoc networks provide the users unparalleled flexibility, they pose serious challenges to the designers. Due to the lack of a fixed infrastruc-ture, nodes must self-organize and recon gure as they move, join or leave the network. All nodes are essentially the same and there is no natural hierarchy or central controller in the network. All functions have to be distributed among the nodes. Nodes are often powered by batteries and have limited communi-cation and computation capabilities. The bandwidth of the system is usually limited. The distance between two nodes often exceeds the radio transmission range, and a transmission has to be relayed by other nodes before reaching its destination. Consequently a network has a multihop topology, and this topol-ogy changes as the nodes move around. Given all these constraints, design of protocols practical for ad hoc networks is often driven by necessity rather than e ciency. At one hand many problems in these networks are inherently di - cult (NP-complete) so people are forced to look for suboptimal solutions; on the other hand the high costs, in terms of the computation and communica-tion overhead, associated with many e cient algorithms limit their practical usages. Among the various aspects of mobile ad hoc networks, medium access control and routing are two most active research areas. The multihop topology allows spatial reuse of the wireless spectrum. Two nodes can transmit using the same bandwidth, provided they are su ciently apart. Many MAC proto-cols, including commercial standards like IEEE 802.11 [6] and HIPERLAN [7], are purely based on contention. These protocols are attractive because they are very simple and easy to implement. They work well under light tra c but su ers from collisions when the tra c becomes heavy. There is another class of protocols based on reservation or scheduling (see [8] and references therein). In these protocols, a transmission schedule is generated rst and nodes transmit and receive according to this schedule. Generation of the schedule sometimes requires substantial overhead, but a good schedule enables more e cient use of the bandwidth and provides better quality-of-service. This is important for these bandwidth-constrained system, especially under heavy tra c. However, many scheduling problems are very di cult (NP-complete) even with accurate information of the entire network. Many works are focused on development of e cient distributed protocols. As real-time multimedia tra c in these networks continues to increase, scheduling-based protocols will become more important.

Another e ective method to increase the capacity of a wireless network is power control [9]. By controlling its transmission power, a node can achieve its trans-mission quality while at the same time reduce the interference in the channel. Power control can also mitigate the effect of nodal movement to some extend. For a network to maximize its capacity, it is necessary that every node carefully adjusts its power, sometimes to the lowest possible level just to reach its nearest neighbor [10]. This is also desirable for increasing the battery life of a mobile node. Although traditionally power control has been studied at the physical layer, in fact it has profound impacts and influences every aspect of the network. Routing is another actively researched area for mobile ad hoc networks. The Mobile Ad-Hoc Networks (MANET) working group of the Internet Engineer-ing Task Force (IETF) has been actively evaluating and standardizing routing, including multicasting, protocols. Because the network topology changes ar-bitrarily as the nodes move, information is subject to becoming obsolete, and di erent nodes often have di erent views of the network, both in time (infor-mation may be outdated at some nodes but current at others) and in space (a node may only know the network topology in its neighborhood and not far away from itself). A routing protocol needs to adapt to frequent topology changes and with less accurate information. Because of these unique requirements, rout-ing in these networks are very di erent from others. Gathering fresh information about the entire network is often costly and impractical. Many routing protocols are reactive (on-demand) protocols: they collect routing information only when necessary and to destinations they need routes to, and do not maintain unused routes. This way the routing overhead is greatly reduced compared to pro-active protocols which maintain optimal routes to all destinations at all time. This is important for a protocol to be adaptive. Often route optimality is secondary to the correctness (loop-freedom) of these routes. AODV [11], DSR [12] and TORA [13] are representatives of on-demand routing protocols presented at the MANET working group. Quality-of-service routing in mobile ad hoc networks is relatively uncharted territory. In order to provide quality-of-service, the protocol needs not only to nd a route but also to secure the resources along the route. Because of the limited, shared bandwidth of the network, and lack of central controller which can account for and control this limited resources, nodes must negotiate with each other to manage the resources required for QoS routes. This is further complicated by frequent topology changes. Due to these

constraints, QoS routing is more demanding than best-e ort routing. What types of QoS are feasible for mobile ad hoc networks and how to achieve them deserve detailed studies.

1.2 Contributions of this dissertation

The problems of transmission scheduling, power control and quality-of-service routing in mobile ad hoc networks are investigated in this dissertation. Gener-ation of TDMA transmission schedules is studied in Chapters Two and Three. Many previous works in TDMA scheduling for ad hoc networks do not consider mobility, and the network size is often fixed and known in advance. This al-lows the scheduling protocol time to gather network information, or to program the information known in advance into the protocol, and to use relatively com-plex schemes to generate e cient schedule. Some works consider distributed algorithms which can be implemented in a real networks, while others do not consider distributed implementation (which is a necessity for a real system) and consider all the network information is known. Nodal mobility makes the prob-lem of TDMA scheduling more di cult, because as nodes move, the network topology changes accordingly. This could cause conflict in the schedules and make information gathered by a protocol obsolete. The network size could also change as the nodes move, join or leave the network. To work in a mobile en-vironment requires a transmission scheduling protocol to generate (or update) a transmission schedule very quickly, so that it can regenerate or update the schedules frequently without incurring too much overhead. The TDMA schedul-ing protocols developed here consider nodal mobility. At the center of their design is the speed with which the schedules are generated (or updated), the way the relevant information of the network is collected and used, and ease of distributed implementation. A Five-Phase Reservation Protocol (FPRP) for generating a broadcast schedule where every node can reserve a time slot to transmit to all its neighbors is developed in Chapter Two. In Chapter Three, we develop an Evolutionary-TDMA scheduling protocol (E-TDMA) for generat-ing general transmission schedules, including unicast (link schedule), multicast and broadcast (node schedule). It speci cally addresses schedule update in the face of network change. These two protocols are fully distributed and require only local information to compute the schedules. They di er from most previous scheduling protocols in that they emphasize the speed with which the schedules are calculated or updated rather than the optimality of these schedules, provided a

reasonably degree of bandwidth e ciency is achieved. The rational behind is that in a mobile network whose topology changes frequently, the cost of calcu-lating a highly e cient transmission schedule is too high to justify its usage.

It is more important to produce a viable schedule with reasonably good e - ciency quickly using a distributed algorithm and limited information. In FPRP and E-TDMA, contention is used for signaling when a node wants to reserve its transmission slot. Because a contention only involves nodes in a two-hop range, it is a local process and does not depend on the size of the network. Use of con-tention makes the protocols both flexible and robust, and a node can reserve the slots it needs quickly, compared with some previous distributed protocols whose overhead grows proportionally with the network size. This way we achieve pro-tocol scalability - they can be used in large networks or networks of changing size. Compared with the standard IEEE 802.11 protocol, in these protocols user data transmission takes place in reserved, conflict-free time slots and provide better spectrum e ciency. They can provide better quality of service, especially under heavy tra c load.

In the development of these two protocols, a simplified network model is as-sumed. The network is represented by its topology which is modeled as a graph. Two nodes are connected by an edge when they are within a predefined trans-mission range. Although widely used, representing an ad hoc network by a graph is only a crude approximation. In fact an ad hoc network does not have a clearly defined topology like most other networks. Because the radio channel is an open medium, in a wide sense a node is connected to every other node of the network. A better representation is to model the network with the propagation gain be-tween every pair of nodes and the attenuation of received signal with distance. Signal-to-interference (noise)-ratio, or SIR, is a more accurate measure of trans-mission quality. In Chapter Four, we study power control and dynamic channel allocation with this more realistic model. Power control in wireless networks has been studied extensively, and a class of simple, distributed yet highly e cient power control algorithm has been developed [14, 15]. In this dissertation we de-velop a distributed channel probing scheme which works in a network applying power control. It allows a node to estimate the channel condition by probing a channel, and to use this information to select its channel dynamically.

Here we deviate a little bit from the context of mobile ad hoc networks, because the channel probing scheme can be applied to a wireless network in general (ad hoc or cellular) applying closed-loop power control, and it can be used for dynamic channel allocation in a TDMA or FDMA network, or for admission control in a DS/CDMA network. Node mobility is not addressed in this chapter. We show that the dynamic channel allocation scheme often found in the literature ([16] for example), where a node chooses a channel with the least interference power, is less accurate; information obtained from channel probing is more accurate and reflects the dynamics of the power control algorithm, and let a node choose a good channel when it is admissible, or blocked from the system before causing much interference when it is not. Compared with some dynamic channel alloca-tion schemes developed early, our scheme both admits more new users and better protects active users. Compared with some channel probing schemes developed earlier which only works for single new user ([17, 18]), our scheme works for the case of multiple new users as well. Di erent links can probe a same channel simultaneously, yet the distributed decisions they make from probing the chan-nel is equivalent to a decision made from knowing the information of the entire network. This makes it a truly distributed scheme.

In Chapter Five, we study quality-of-service routing in mobile ad hoc net-works. It is di cult to provide QoS in a large network where the topology

changes very frequently, but in a small network where the topology changes at a relatively slow rate, we try to provide QoS for sessions transmitting at constant bit rate by establishing routes with reserved bandwidth. The QoS measure is the amount of bandwidth a flow enjoys on its route given in number of time slots, assuming TDMA is used at the MAC layer. The key to provide QoS is the ability to manage the network resources, which is TDMA transmission time slot here. Some QoS routing protocols developed early for ad hoc networks use abstract notions for resources and do not reflect the need for conflict-free TDMA transmissions precisely. They often ignore the interference between dif-ferent transmissions. QoS routing in a TDMA-based ad hoc network has not been studied previously. It is more challenging than other types of networks because di erent transmissions can interfere with each other. We begin with the problem of

accounting for the resources in a TDMA-based mobile ad hoc network and rst study how to calculate the end-to-end available bandwidth on a route. We show it is NP-complete to nd the maximum bandwidth for a route. We then develop an e cient distributed scheme for bandwidth calculation. By integrating this bandwidth calculation scheme with the AODV routing protocol, we develop a QoS routing protocol for mobile ad hoc networks. This protocol can nd and maintain a QoS route satisfying the bandwidth requirement in the presence of node movement. Compared with the original, best-e ort AODV protocol, the QoS routing protocol not only provides QoS to individual flows, but also achieves load balancing and route redundancy. Simulations show that it increases the network throughput and decreases the packet delay, especially under heavy tra c condition. This QoS routing protocol mainly applies to small networks or short routes under low node mobility. For a large or highly mobilenetwork, it lacks the scalability and the flexibility to deal with frequent route failures. How to provide QoS in large networks needs further investigations.

Chapter 2

A Five-Phase Reservation Protocol (FPRP) for broadcast scheduling in mobile ad hoc networks

2.1 Introduction

We consider the problem of scheduling TDMA broadcast in a mobile ad hoc network. The multihop topology of an ad hoc network allows spatial reuse of the bandwidth. Di erent nodes can use the same bandwidth simultaneously as long as they are su ciently separated and do not interfere with each other. The problem of assigning the transmission time slots to the nodes is referred to as scheduling. Here, we consider the problem of scheduling broadcast transmissions in a single channel radio network where nodes employ omni-directional antennas. By broadcast, we mean that when a node transmits, every one-hop (adjacent) neighbor of the node receives the packet. A broadcast schedule is very useful to have in a network's control/organization phase, where nodes need to coordinate control actions with each other. Here, a conflict-free broadcast schedule requires that any two simultaneously transmitting nodes be at least three hops apart. Many algorithms have been developed to schedule broadcasts in multihop radio networks [19,20,21,22,23,24,25,26,8,27,28,29,30,1]. Some of them are centralized algorithms and depend on a central controller [24,8] with global knowledge to generate the schedule for entire network. These algorithms can generate schedules with good bandwidth e ciency. However, it takes a lot of overhead for the controller to gather information about the entire network, and in the presence of nodal mobility, this information may be obsolete. It is also computation-intensive for the controller to generate the schedules, and the central controller is a single point of failure. Some schemes are distributed, and one by one nodes reserve their transmission slots following some fixed or-der [20,22,25]. Consequently the length of the scheduling process grows with the size of the network. These protocols also require the nodes to have some a prior knowledge about the network (such as size and membership). Hence, these protocols are inappropriate for large networks or networks of varying size. The focus of some recent work is to generate topology-transparent TDMA schedules ([29,30,31]). These schedules are independent of the speci c topology and therefore immune to nodal mobility. This makes them particularly attractive to mobile ad hoc networks. However, the bandwidth e ciency of a topology-transparent is lower than that of a topology-dependent schedule due to the in-herent redundancy in order to work topology-independently. E cient operation of these schedule also requires an instant feedback

channel. Because such an instant feedback channel may not be available in a real ad hoc network, the topology-transparent schedules are not alway applicable.

A new single channel, time division multiple access (TDMA)-based broad-cast scheduling protocol, termed the Five-Phase Reservation Protocol (FPRP), is developed in this chapter. The protocol jointly and simultaneously performs the tasks of channel access and node broadcast scheduling. The protocol allows nodes to make reservations within TDMA broadcast schedules. It employs a contention-based mechanism with which nodes compete with each other to ac-quire broadcast TDMA slots. The FPRP is free of the \hidden terminal" prob-lem, and is designed such that reservations can be made quickly and e ciently with negligible probability of conflict. It is fully-distributed and concurrent, and is therefore scalable. A \multihop ALOHA" policy is developed to support the FPRP. This policy uses a multihop, pseudo-Bayesian algorithm to calculate contention probabilities and enable faster convergence of the reservation proce-dure. The performance of the protocol, measured in terms of scheduling quality, scheduling overhead and robustness in the presence of nodal mobility, is studied via simulations. The results showed that the protocol works well in all three aspects.

2.2 The FPRP Protocol

2.2.1 Overview

The FPRP is a contention-based protocol which uses a ve-phase reservation process to establish TDMA slot assignments that are non-conflicting with high probability. The FPRP is fully-distributed protocol and executes in parallel over the entire network. By parallel, we mean that the FPRP permits multiple reservations to be made at various parts of the network simultaneously. The reservation process for a given node only involves nodes within a two-hop radius, and is thus a local process. No coordination is necessary with more distant nodes. By keeping the reservation process localized and running simultaneously over the entire network, the FPRP is insensitive to the network size. This makes the protocol suitable for a large network, or a network whose size changes dynam-ically. It also works e ciently when the network becomes partitioned. A node needs no *a priori* information about the network, i.e. it does not need knowledge regarding network membership, its neighbor set or network size. This makes the FPRP robust in a frequently-changing topology. The FPRP does not need the support of

additional protocols for medium access control or network explo-ration. The protocol jointly and simultaneously performs the tasks of channel access and node broadcast scheduling. A node uses the FPRP to explore its neighborhood and to make nearly conflict-free reservations. The FPRP has no restriction on the topology of the network, except that it requires that every link is bidirectional. The topology can be represented by an undirected graph.

A major di culty in a wireless environment is the \hidden terminal" prob-lem [32]. Due to the limited range of wireless transmissions, two nodes can be far enough apart that they cannot detect each other directly (they are \hidden" from each other), yet their transmissions may collide at another node in the mid-dle. Even the four-way handshaking scheme used in IEEE 802.11 cannot prevent collisions completely [6, 33]. In FPRP, the collision from two hidden nodes is detected at the node where it occurs, and it is up to this node to explicitly inform both transmitters. This ensures that no collisions due to hidden nodes can arise in the TDMA broadcast schedule.

2.2.2 Assumptions

We make the following assumptions regarding the networking context in which the FPRP operates:

Nodes keep perfect timing. Global time is available to every node, and is su ciently tight to permit global slot synchronization; A link between two nodes is symmetric, i.e. two nodes either talk to each other perfectly, or do not interfere at all. Packet collision is the only sourceof receiving error; During the interval in which the FPRP is performed, the topology of the network does not change. The rationale for this is that the network's topology is slowly changing relative to the time required to compute a new transmission schedule. Also, the nodes may move around, but the speed with which they move is slow compared with number of times a transmis-sion schedule may be used. Thus, once a TDMA schedule is computed, it can be used for some time before a topological change forces another schedule (or an update) to be made; When multiple packets arrive at a node, all of them are destroyed (i.e. no capture); A node is able to tell whether zero packet, one packet, or multiple packets were transmitted, provided that it is in receiving mode itself;

Every node has a unique ID.

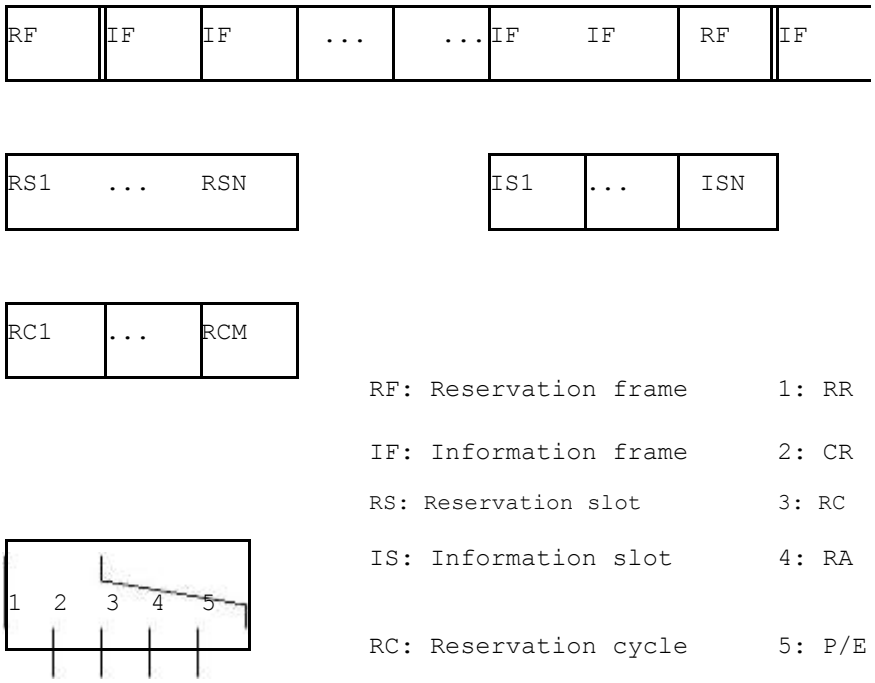


Figure 2.1: Frame structure of the FPRP.

2.2.3 Detailed Description

The protocol's frame structure (shown in Figure 2.1) is as follows. There is a Reservation Frame (RF) followed by an information epoch, which consists of a sequence of Information Frames (IF). There are *N* Information Slots (IS) in an IF. There are also *N* Reservation Slots (RS) in an RF. Each RS is dedicated to the reservation of a corresponding IS. If a node wants to reserve an IS, it contends in the corresponding RS. A TDMA schedule is generated in the RF, and is used in each of the subsequent IF's until the next RF, where the schedule is regenerated.

A RS is composed of *M* Reservation Cycles (RC) (the value of the parameteris determined heuristically). Each RC consists of a ve-phase dialogue from which the protocol receives its name. Within a RS, a reservation is made through a sequence of ve-phase dialogues between a contending node and its neighbors loosely stated, a node that wishes to make a reservation rst sends out a request, and feedback is provided from its neighbors regarding the request. If the

request is successful (i.e. it does not collide with other requests), the node reserves the slot. This reservation information is passed to every node within two hops. These nodes will honor this reservation and will not contend further for the slot. If not successful, the node will contend in subsequent RC's for this RS with some probability until itself, or another node one or two hops away, succeeds. As a result, the node will either transmit (T), receive (R) or be blocked (B) in the corresponding information slot. The ve-phase dialogue ensures: 1) if two requests collide, neither makes the reservation; 2) once a node makes a reservation, it will have sole use of the slot in its neighborhood with high probability. As will be seen, the design of the protocol allows a slot be spatially reused e ciently throughout the network.

The Five-Phase Dialogue

A node keeps global time, and knows when a ve-phase cycle starts. A node can transmit or receive, but cannot do both at the same time. We assume every node participates in the reservation process.

A reservation cycle has five phases. They are:

1. Reservation Request phase (RR), where nodes make their requests for reservations;
2. Collision Report phase (CR), where nodes report collisions that just oc-curred in phase 1;
3. Reservation Con rmation phase (RC), where nodes make con rmations of their requests (a reservation is established in this phase);
4. Reservation Acknowledgment phase (RA), where nodes that heard a RC in phase 3 acknowledge with a RA packet. This RA also serves to inform those nodes that are two hops away of the recent reservation;
5. Packing and Elimination phase (P/E). In this phase, two kinds of packets are transmitted. A packing packet, which serves to make the broadcasting pattern denser in a given slot, and an elimination packet, which is used to remove possible deadlocks (DL) between adjacent broadcast nodes.

The first three phases are analogous to the distributed protocol in [25].

The details of each phase are given below:

1. Reservation Request Phase: In this phase, a node which wants to make a reservation sends a Reservation Request packet (RR) with probability p. A node sending a RR is referred to as a Requesting Node (RN). The calculation of the probability p will be discussed later. A node which does not transmit a RR listens in this phase. It may receive zero, one or multiple RR's from its neighbors. In the last case, we assume that all the RR's are destroyed and the node senses a collision. A node does not need to tell how many packets are involved in a collision.

2. Collision Report Phase: If a node receives multiple RR's in phase 1, it transmits a Collision Report packet (CR) to indicate the collision. Other-wise it is silent. By listening for any CR's in this phase, a RN determines whether its RR has collided with others. On receiving no CR, it assumes that its RR reached every neighbor safely. Such a node then becomes a transmission node (TN). It will go ahead and make a reservation in phase 3, and transmit in the subsequent information slots unless disabled in phases 4 or 5.

A CR packet is a form of negative acknowledgment (NACK). Receiving one or more CR packets indicates a failure; only no CR indicates a success. It should be clear that if two RN's are hidden from each other, their RR's would collide in the middle and both would receive the CR. No reservation is made. Thus the RR/CR exchange eliminates the \hidden terminal" problem. Reservation Con rmation Phase: A TN sends a Reservation Con rmation packet (RC) in this phase. Every node which is one hop away receives the RC and understands that the slot has been reserved. They will receive from the TN in the corresponding information slots. They will not contend further for this slot.

3. Reservation Acknowledgment Phase: In phase 4, a node acknowledges a RC it just received by sending a Reservation Acknowledgment packet (RA). This tells a TN that its reservation has been established. If a TN is not connected with any other nodes, it does not receive any RA and thus becomes aware of its isolation and no longer considers itself as a TN. This prevents isolated nodes from transmitting and wasting its energy. Without this phase, an isolated RN would never receive a CR and would then always become and remain a TN.

A RA transmission also serves to inform the nodes which are two hops away from a TN of the success. These nodes also label this slot as reserved and cease contention. They become blocked (B) in this slot.

We de ne transmitter deadlock (DL) to be the situation where two or more TNs are adjacent|these nodes are referred to as \deadlocked nodes". Transmission nodes involved in a deadlock do not share a common neighbor which itself is not a TN, for the reason we will soon see. Deadlocks begin to form during phase 1. Because nodes cannot receive while transmit-ting in phase 1, they cannot sense a collision directly. To avoid deadlock, the transmitting nodes must rely on the existence of a *common* neighbor to send a CR in phase 2. If no such neighbor exists|in the absence of a CR|they will each claim success, become TNs during phase 2 and a deadlock is formed. A TN involved in a deadlock can be of one of two types: (i) an isolated deadlocked node when it is not connected to any other, non-deadlocked node, and (ii) a non-isolated deadlocked node when it is connected to an adjacent, non-deadlocked node. Phase 4 also serves to resolve deadlocks involving an isolated deadlocked node. Since an isolated deadlocked node does not have any neighbor which is not a TN itself, it will not receive a RA in Phase 4. On hearing no RA, it will abort its trans-mission thus resolve the deadlock. This will not happen to a non-isolated deadlocked node, because a non-isolated deadlocked node will receive at least one RA from its neighbors. Probabilistic resolution of non-isolated deadlock is performed in phase 5. Packing/Elimination Phase: In this phase every node that is two hops from a TN which has made its reservation since the last P/E phase sends a Packing Packet (PP). A node receiving a PP therefore learns there is a recent success three hops away. As a consequence, some of its neighbors cannot contend further for this slot. It can take advantage of this and adjust its contention probability accordingly (Section 2.3). This can speed up the convergence. It also increases the success probability of nodes that are three hops away from other nodes already possessing a reservation in this slot. Hence, two TN's are more likely to be only three hops apart rather than further. This is preferable, because, when TN's are only three hops apart, more nodes are allowed to transmit and less nodes are blocked. This can be called \maximal packing". Through the encouragement of maximal packing, the FPRP uses a slot more e ciently.

In the same phase, each TN sends an Elimination Packet (EP) with a probability of 0:5. This is intended for another TN, which could be poten-tially adjacent, in an attempt to resolve a non-isolated deadlock. If a TN does not transmit, but receives an EP in this phase, it learns there is a deadlock. In this case it will relabel the slot as reserved by the other TN (the one that sent the EP) and will receive, rather than transmit, in the slot. It will contend further in other slots. There is no need to inform its neighbors about this relabeling event.

Additional EP's can be sent in order to further reduce the probability of deadlock. This can be achieved if a TN, after acquiring a reservation, transmits an EP in phase 1 of every cycle in the same reservation slot. This EP will not interfere with any RR's (after a reservation is made, every node within 2 hops will not contend in the same slot, so the EP from the TN cannot collide with a RR). An EP in phase 1 works in the same manner as an EP in phase 5. The elimination process is thus executed more often and our simulation results showed that the remaining DL probability becomes negligible.

The fifth phase helps only after a successful reservation is made. Since the throughput of contention-based protocols (such as slotted ALOHA) is much lower than one packet per slot, it is more economical to place a fifth phase in every few reservation cycles. Thus, a sequence would be a sequence of one, two or three four-phase cycles followed by a fifth phase. How often a fifth phase is used can be determined heuristically.

The five-phase scheme attempts to minimize the probability of collision in a way that is e cient and robust. The meaning of a packet is implicitly conveyed simply by *when* (i.e. in which phase) the packet is sent. Thus, a packet need only consist of a single, logical bit. A packet may collide with another packet, but the correct semantic is always inferred in the context of the protocol. The decision is made on the basis of the absence/presence/collision (0=1=*e*) of various packets. A packet needs no more than a logic bit. In fact, this logic bit needs to be long enough such that a receiver can distinguish between 0, 1 and *e*. The packets can be made very small, thus a reservation cycle is very compact. The FPRP uses the fact that a collision always occurs one hop away from the sender. A collision is detected at the node where it occurs (unlike the CSMA/CA protocol, where the sender detects the collision at the receiver) and is signaled to the sender which functions as a local hub. It collects collision information and makes the nal decision. Before a reservation is deemed successful, no information has to be collected from or dissipated to nodes more than one hop away. This greatly simpli es the reservation process.

It is worth mentioning that the five {phase scheme has many elements that are similar to other existing MAC protocols. The first four phases bear a resemblance to the popular RTS-CTS exchange [6]. The elimination mechanism in phase 5 is similar to that used in HIPERLAN [7]. Each elimination packet is an elimination process of one bit. The protocol requires 0=1=*e* detection by the physical layer. In phase 1, it is necessary to di erentiate between 0, 1, or *e*. In the

other phases, it is only necessary to tell whether it is 0 or not. The major di erence is that the FPRP is a synchronous protocol and requires tight timing.

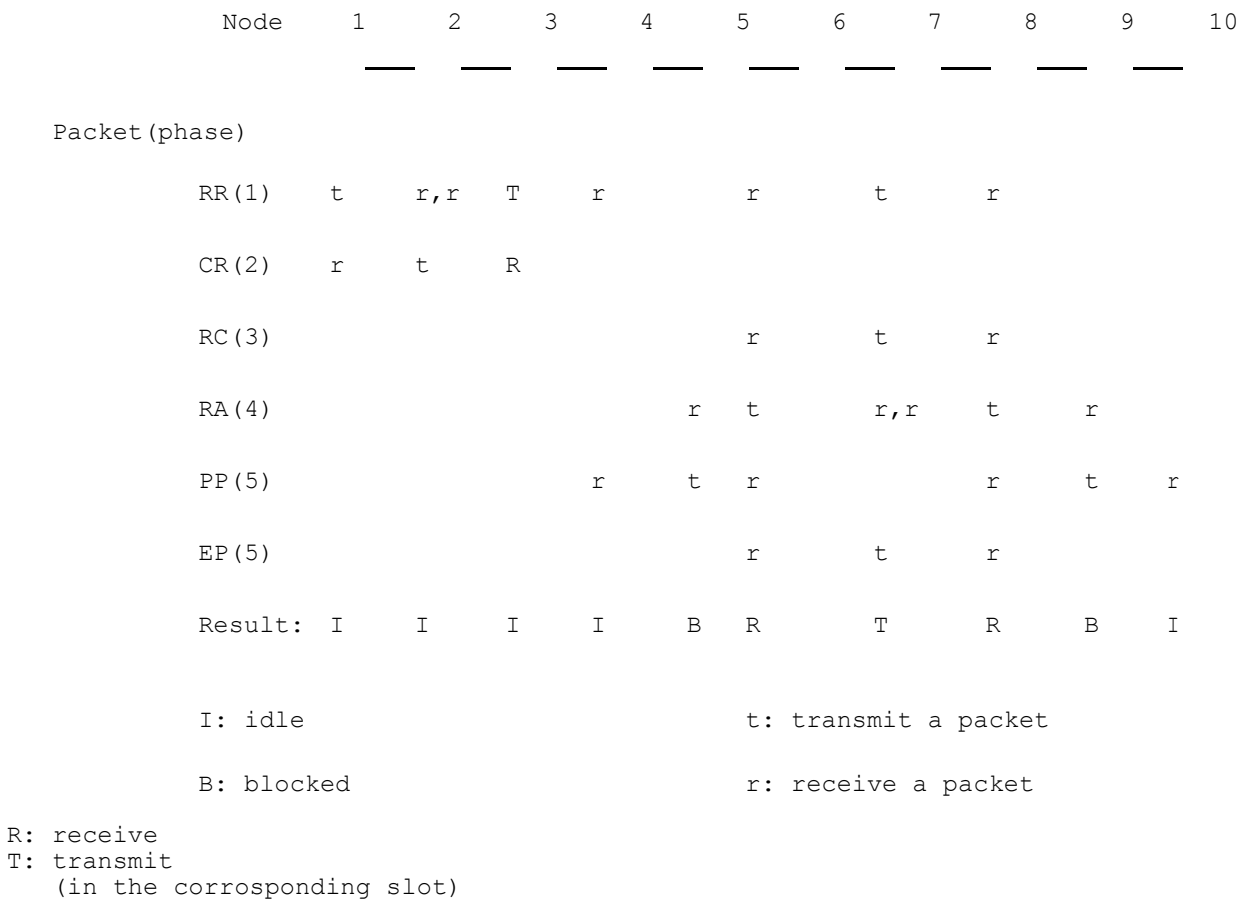


Figure 2.2: A five-phase reservation cycle in a tandem network.

2.2.4 Examples

We illustrate the execution of a ve-phase cycle within a tandem network of 10 nodes (see Figure 2.2).

No reservations have been made before this cycle. A vfie-phase cycle is shown, along with the transmission of every node in each phase. In phase 1, nodes 1, 3 and 7 transmit RR's. The RR's from nodes 1 and 3 collide at node 2, while the RR from node 7 reaches its neighbors (nodes 6 and 8) ungarbled. In phase 2, node 2 reports the collision. On hearing the CR from node 2, nodes 1 and

3 become aware of the collision and do not proceed further. Node 4, which receives a RR in phase 1 but nothing in phase 3, learns that the RR from node 3 collided with another RR somewhere else. Node 7 does not receive any CR from its neighbors and assumes there is no

collision. So, in phase 3, it sends a RC telling nodes 6 and 8 of its con rmation of its reservation. In phase 4, nodes 6 and 8 acknowledge with RA's. Their RA's also inform nodes 5 and 9, which are two hops away from 7 that a successful reservation was just made and they are blocked from contending further in the following cycles for the same slot. In phase 5, node 7 transmits an EP. Note that there is no deadlock in the example and this EP eliminates nobody. (In reality, DL's are more likely to occur in a tandem network because every link is a \bridge". The elimination procedure is most important in a network like this). Simultaneously, in phase 5, nodes 5 and 9 transmit PP's announcing the recent success of node 7, thereby encouraging nodes 4 and 10 to contend. By adjusting their contention probability (to be discussed in Section 2.3), nodes 4 and 10 become more likely to succeed in the following cycles.

The previous example illustrates the mechanism of the FPRP. However, a real ad hoc network rarely has a linear topology|it is more likely to be a \mesh". Such a network with 16 nodes is shown in Figure 2.3. With this example, we wish to emphasize the fact that the FPRP is parallel. The algorithm runs in parallel on every node, and multiple reservations can be made simultaneously at di erent parts of the same network. Figure 2.3 shows the rst four phases of the FPRP. In phase 1 (a), 5 nodes (1, 4, 5, 11, 15) send out RR's. Among them, the RR's from nodes 4 and 5 collide at node 6, which transmits a sole CR in the next phase (b). This CR rejects the requests from nodes 4 and 5. The other RN's (1, 11, 15), hearing no CR, con rm their reservations in phase 3 and become TN's (c). In the rst three phases, the transmission ranges of various packets are shown with circles. It can be seen in this phase that the transmission ranges of these TN's do not overlap, i.e. no collision occurs. In phase 4 (d), the reservations are further relayed to all nodes two hops away. The enclosed area of Figure 2.3.d shows every node that is a ected by the phase 4 transmissions.

In this example, after a reservation cycle, three reservations are established by nodes 1, 11 and 15. These nodes are at least three hops apart and do not mutually interfere. More nodes would make reservations in the same cycle in a larger network, and this number grows proportionally with network size.

The operation of the FPRP can be summarized as follows. The rst four phases are used to establish reservations and eliminate the hidden terminal prob-lem. The fth phase performs packing and elimination in order to make more e cient spatial reuse of the same slot and to eliminate deadlocks that may exist between adjacent nodes.

2.2.5 Correctness

A broadcast is successful only if every neighboring node receives the packet successfully. A node cannot receive packets from more than one sender, nor can it receive and transmit simultaneously. A node receives a packet successfully only if the packet is the only one it receives, and the node itself is not transmitting at the same time. We call the collision of packets at a node which is not transmitting a type I collision, and the collision of packets at a node which is transmitting a type II collision. The hidden node problem is a special case of the type I collision. A type II collision where the TNs do not have a common neighbor is the same as a deadlock.

Proposition 2.1: A type I collision cannot happen.

Proof: When more than one RR's reach a node at the same time, if this node is not transmitting, it senses the collision and transmits a CR. All the adjacent RN nodes receive the CR and none of them succeeds. If a TN is the rst one to make a successful reservation, every other node within two hops is informed (in phase 3 for one-hop neighbors and in phase 4 for two-hop neighbors). These neighboring nodes will honor the reservation and will not contend further in the same slot. So once such a reservation is made, it will be the only one in its neighborhood. It can be concluded that no two transmissions would collide at a third node, i.e. a collision of type I cannot happen. Q.E.D.

Claim 2.1: A type II collision can only happen with very small probability.

Justi cation: A type II collision is always resolved when two adjacent RNs share a common neighbor, which is often the case in a mesh shaped network. If two neighboring nodes request at the same time and they do not have a common neighbor, neither will discover the collision. If one of the TNs is isolated, it does not have a neighbor which is not a TN and will not hear a RA in Phase 4. It will abort its transmission and the deadlock can be resolved. For a deadlock that cannot be resolved this way, all the TNs will reserve the same slot. Our simulations showed that

deadlocks involving more than two nodes are very rare. A deadlock is most likely to form at a bridge" (A bridge is a link between two larger groups of nodes that are otherwise not locally connected. The nodes at either end of the bridge do not share any common neighbors). When such a DL is formed, the adjacent TN's use elimination packets in an attempt to eliminate each other. After every subsequent elimination phase (and embedded elimination in Phase 1), the probability of a deadlock is reduced by half. Simulation results, to be discussed shortly, indicate that deadlock is likely to be resolved during the elim-ination process, especially if this is embedded in phase 1 as described previously. Based on these results, we conclude that the probability of a type II collision is very small, and it does not signi cantly a ect the performance of the FPRP.

2.2.6 Application to Graph Coloring

The graph coloring problem corresponding to the TDMA broadcast slot assign-ment problem is well known [22]. It consists of assigning colors to the nodes of a network such that no two nodes within two hops of each other have the same color. This can be transformed to the standard graph coloring problem as follows. For a given graph $G(V; E)$ (with a set of nodes V and a set of edges E), if we connect every pair of nodes that are two hops apart, we get a new graph G^0. The graph G^0 is called the \square" of the original graph G. The problem be-comes how to color G^0 so that the same color is not given to adjacent nodes. The problem of coloring a graph with the minimal number of colors is NP-complete [34]. Various heuristics have been developed. Recently it was shown that global sorting of some kind produces good results [8]. Among them is the RAND pro-tocol, where nodes are colored in a random ordering in a greedy fashion. In fact, the RAND algorithm is used in many channel assignment schemes, and its performance is well studied and documented [8]. Therefore we use the RAND algorithm as a benchmark. We now evaluate the performance of the FPRP when used as a pure graph coloring protocol (i.e. one that assigns one slot or color to every node). We also compare its performance with the RAND protocol and a degree-based lower bound. This degree lower bound is the maximal degree of the graph plus one. This lower bound is found to be very tight, and is used to approximate the optimal coloring solution.

Simulation Results

Networks with random topologies are generated as follows. For a network of size N , N nodes are generated in an area of N by N units. The location of a node is generated randomly, using a uniform distribution for its X and Y coordinates. Thus the average density of the network is 1 node per square unit. The transmission range R of a node is chosen typically to be 1:5 units. The purpose of generating a network this way is so that the size of the network and the transmission range R (relative to the node density) can be varied indepen-dently. The transmission range is the same for every node, making every link bidirectional. The average degree of a node is approximately 7. The generated network is converted into a undirected graph G (V; E). The FPRP and RAND protocols are used to color the graph. In the FPRP, every node stops contention after it acquires a color. During each cycle, some nodes acquire the correspond-ing color. The reservation cycles are repeated until the FPRP converges, e.g. the same color can not be assigned to any other nodes in the graph. The next color is assigned with the same fashion. The FPRP terminates after every node has acquired a color. The number of colors required is the measure of coloring (scheduling) quality.

Networks of various sizes ranging from N = 100 to N = 500 are tested. The transmission range of R = 1:5 is used for all of them. The results are given in Table 2.1. DLB is the degree lower bound. The effect of increasing connectivity (R) on a given network is also investigated. A network of 100 nodes is produced and the transmission range R varies from 1:0 to 3:0. As the number of neighbors increases, so does the number of colors used. The results are shown in Table 2.2. The overall performances of the FPRP and the RAND are comparable, and they are only slightly higher than the degree lower bond. Essentially, both are randomized coloring processes and they are expected to perform similarly. It is worth noting that while the RAND algorithm is a centralized solution and requires global knowledge as to which nodes have been given what colors (distribution versions are available and the global knowledge can be acquired gradually); the FPRP, on the other hand, is totally distributed and fully parallel requiring no *a prior* knowledge. This makes the FPRP more practical and more implementable on a large, mobile ad hoc network.

Size	DLB	RAND	FPRP
100	15	16	16
200	16	19	17
300	15	17	17
400	15	18	19
500	19	21	22

Table 2.1: Coloring of networks of di erent size. The transmission R = 1:5.

R	DLB	RAND	FPRP
1.0	9	9	9
1.5	15	16	16
2.0	20	23	24
2.5	29	32	33
3.0	33	39	38

Table 2.2: Coloring of networks of di erent transmission range R. The number of nodes N = 100.

2.3 Contention-based Access

2.3.1 Rivest's Pseudo-Bayesian Algorithm

The FPRP requires a suitable contention policy. Theoretically, since every node has only one packet to send in a reservation frame, any slotted ALOHA policy can be used as the contention process would always be stable. However, a good policy would make the reservation process converge quickly. Most ALOHA protocols are developed for networks with a central basestation [35]. The situation here di ers in that it is a *multihop* environment and there is no basestation. Every node is a potential source or destination of a packet. We are not aware of a protocol that perfectly meets this requirement. Therefore, we choose to modify Rivest's pseudo-Bayesian Broadcasting Algorithm [36] to t this role.

In Rivest's pseudo-Bayesian algorithm, every node estimates the number of contenders (n) and adjusts its contention probability $p := 1=n$. After every contention slot, a node updates its estimate n on the basis of the feedback: success or idle

$$n := n - 1; \tag{2.1}$$

collision

$$n := n + (e - 2)^{-1}. \tag{2.2}$$

It is designed to support stable throughput with minimal amount of delay. The original algorithm works for a single-hop ALOHA network fairly well. The situation here di ers in that: 1) a node only cares for the contenders which are within two hops of itself; 2) the network typically has a random shape and every node has di erent neighbors; 3) every node has only one packet to send; 4) in the contention for a particular slot, if a node succeeds, every other node within its two hop range will not contend further for the same slot, but will resume contention in other slots. Here we transform Rivest's algorithm into a multihop, pseudo-Bayesian algorithm to adapt to these characteristics.

2.3.2 Multihop Pseudo-Bayesian Algorithm

A node estimates the number of contenders within two hops and calculates its contention probability accordingly. From a node's point of view, n is the number of contenders within two hops of itself. They are called \neighboring contenders". A node updates its estimate on the basis of what it hears: success A node always learns of a success within two hops, for it is either informed in phase 3 (in Figure 2.2, nodes 6 and 8 are informed of node 7's success) if the success is one hop away, or in phase 4 (nodes 5, 9) if the success is two hops away. In the Packing phase, a node learns of a recent success three hops away (nodes 4, 10). Idle an idle is always detected (if there is no node contending within its two hop range, a node hears nothing and thus assumes the slot is idle). Collision Detecting a collision is more complicated. A node knows of a failed contender which is one hop away. If it receives more than one RR (node 2), it senses the collision directly. If it receives a RR in phase 1 but no RC in phase 3 (node 4), it reasons that there is a node contending one hop away and its RR has collided. If a node receives no RR in phase 1, but receives a CR in phase 2, it knows that two nodes which are two hops away are contending and that their RR's collided at one of its immediate neighbors. A collision two hops away cannot always be detected. In the example (see Figure 2.2), node 5 does not know that node 3 contended and collided with node 1. This occurs when one of the contenders is two hops

away, while the other is three or four hops away. In the current protocol, a node has no way to detect a collision like this and we conjecture that theoverhead required to detect such collisions is not worth the cost. We opt to ignore these cases at this time.

If a node hears a success within two hops, it will stop contention in the same slot but will contend in other slots. This results in an oscillation of the number of contenders in a neighborhood. To maintain a stable throughput (success rate), a node needs to keep two estimates: one for the number of nodes that contend within two hops, n_c; the other for the number of nodes within two hops which need reservations, but cannot contend in the current slot due to a nearby success, n_b. Some heuristic constants are used to estimate the effect of a success on the number of contenders nearby. The effect of a success on its neighbors is modeled as follows: for a node one hop away from the success, a portion (R_1) of its neighboring contenders cease to contend in the current slot; for a node two hops away, this ratio is R_2; and for three hops away, R_3. The pseudo-Bayesian algorithm becomes:

1. At the beginning of a reservation slot, a node resets its n_c and n_b as follows:

$$n_c := n_c + n_b; \quad n_b := 0: \tag{2.3}$$

(For the very fi*rst* reservation slot, $n_c := n_{c0}$, where n_{c0} is a prede ned constant.)

2. After every reservation cycle, on hearing an: idle

$$n_c := n_c - 1; \tag{2.4}$$

Collision

$$n_c := n_c + (e - 2)^{-1}; \tag{2.5}$$

success if the success is some x hops away, where x is: zero (itself is the successful node): done; one (it does not contend in the same slot anymore):

$$_c := n_c - 1; \tag{2.6}$$

$$_b = n_b + n_c \quad R_1; \tag{2.7}$$

$$= n_c \quad (1 - R_1): \tag{2.8}$$

c

Two (it does not contend in the same slot anymore):

$$n_c \quad := n_c - 1; \tag{2.9}$$

$$n_b \quad := n_b + n_c \quad R_2; \tag{2.10}$$

$$n_c \quad := n_c \quad (1 - R_2): \tag{2.11}$$

Three:

$$n_b \quad := n_b + n_c \quad R_3; \tag{2.12}$$

$$n_c \quad := n_c \quad (1 - R_3): \tag{2.13}$$

3. It then calculates the contention probability $p := 1=n_c$; if it is able to contend in the next cycle, it contends with probability p.

It needs to be pointed out that this is a heuristic scheme and is not optimal by any means. Even if a node knows the number of active contenders n_c in its two hop range exactly, its contention probability $p := 1=n_c$ is optimal only when every node within its two hop range contends with the same probability. More often than not, di erent nodes have di erent n_c, and each calculates its contention probability based on its own n_c. R_1, R_2 and R_3 can be evaluated with Monte Carlo simulations, or for some cases, calculated analytically. However, because the number of contenders in a neighborhood does not increase, stability is not an issue for the contention process.

Simulation Results

The multihop, pseudo-Bayesian algorithm described above is implemented and tested in the graph coloring process as described in Section 4 for the same network. The parameters, R_1, R_2 and R_3 are evaluated with Monte Carlo sim-ulations in the networks described early. In the simulations presented here, $R_1 = 0{:}80$, $R_2 = 0{:}60$ and $R_3 = 0{:}33$. The number of FPRP cycles required for the protocol to converge for each color is used to study the speed with which the reservations are being made. The network size N varies from 100 to 400, and the transmission

range R = 1:5 for all of them. The simulations were performed 100 times and the results were averaged. The results are shown in Figures 2.4 and 2.5. The multihop pseudo-Bayesian algorithm converges steadily and fast. The number of FPRP cycles (Figure 2.4) used only increase slightly when the network grows from 100 nodes to 400 nodes. A closer look showed that the to-tal number of FPRP cycles increases with the network size approximately as a logarithm function. The total number of FPRP cycles is a measure of schedul-ing overhead. With this logarithmly growing overhead, the FPRP protocol is scalable and is applicable for large networks. Figure 2.5 shows the number of transmitting nodes in each slot, and they grow proportionally with the network size. When we normalize the transmitting nodes with the network size, all the curves in Figure 2.5 agree very well. This implies that the bandwidth e ciency of the schedules (scheduling quality) does not vary with the size of the network.

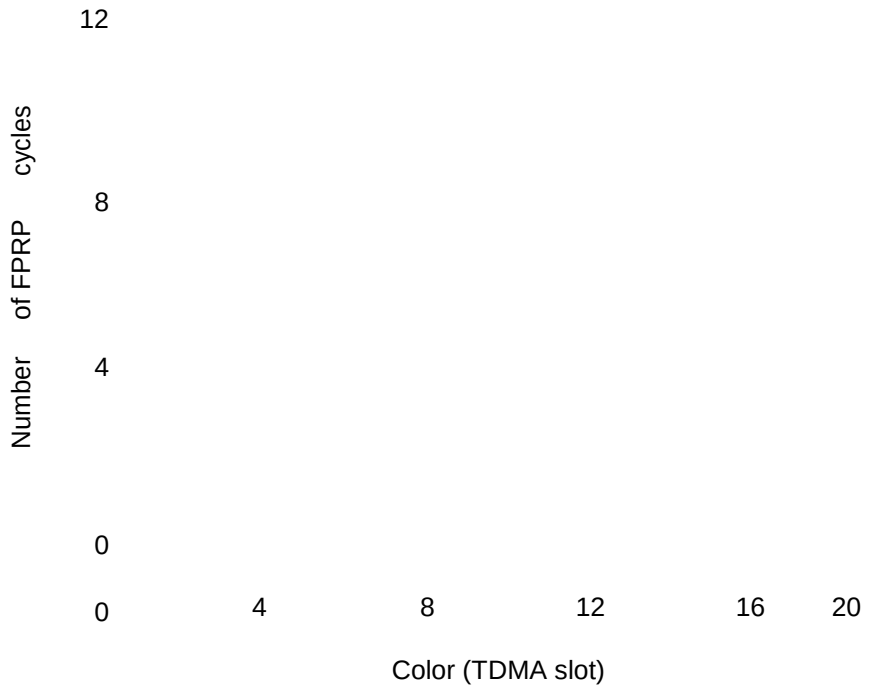


Figure 2.4: Average number of FPRP cycles used to assign nodes to each color (TDMA slot). The transmission range R = 1:5. The network size is 100, 200, 300 and 400, respectively (from bottom to top). with the total number of reser-vation cycles 89, 116, 130 and 145. A closer look showed the overhead increases logarithmly with the network size.

In the simulation, a coordinator is used to globally monitor the coloring process to determine when all the nodes are colored. However, use of such a coordinator is infeasible in a real network. It is possible, based on the simula-tions, to predict how many cycles are necessary once

the typical topology (nodal density and transmission range) is known. On average, it takes between 4.2 (N = 100) and 6.9 (N = 400) cycles to assign a transmission slot to the nodes. We nd that if we use 8 FPRP cycles to assign every color, and 21 colors in total, every node will have a probability higher than 0:99 of obtaining one of the colors. Figure 2.6 shows the number of nodes assigned to each color when these fixed parameters are used. Compared with the case when the global convergence is monitored, the number of transmission nodes becomes \heavy-tailed". A fur-ther increase in the number of FPRP cycles and in the number of total colors would drive this probability very close to 1, but the gain is not likely worth the cost in scheduling delay. Once known, these parameters can be built into the protocol.

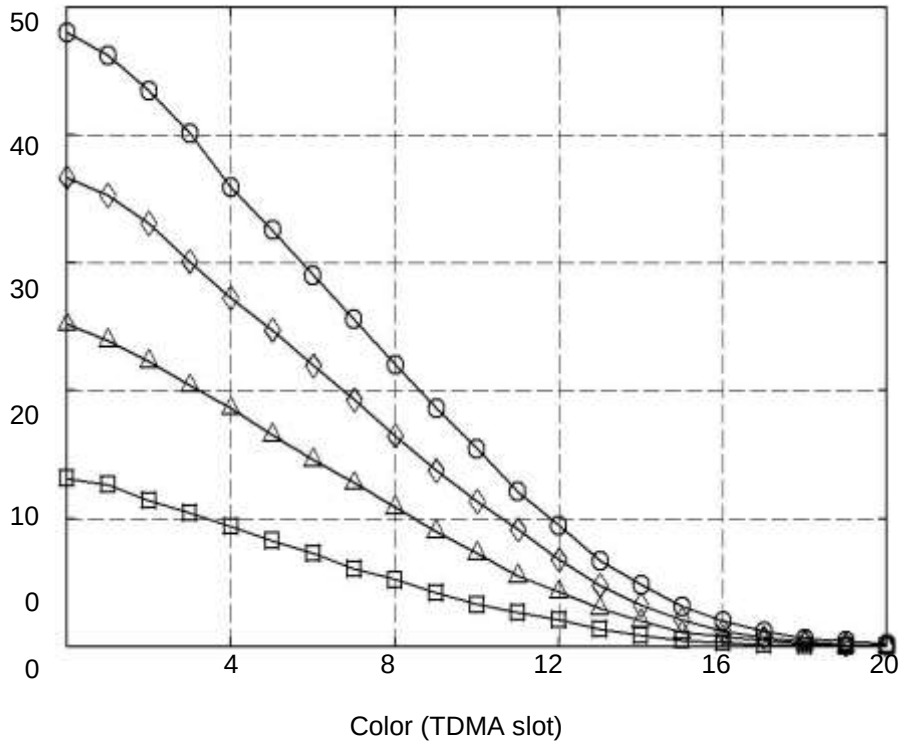


Figure 2.5: Average number of transmission nodes assigned to each color. The network size is 100, 200, 300 and 400, respectively (from bottom to top). They agree very well when normalized with network size.

This permits protocol execution which needs no coordination at all. From a node's point of view, it knows how many colors are available, and which cycle is for which color. It simply uses the FPRP to acquire a color. The simulations also showed that cases of non-isolated deadlock almost never occur. Most of the deadlocks are resolved by the elimination process, and the residual

collision probability is about 0.001. It is reasonable to conclude that the colli- sion probability of the FPRP is very small and has no signi cant effect on the performance of the protocol.

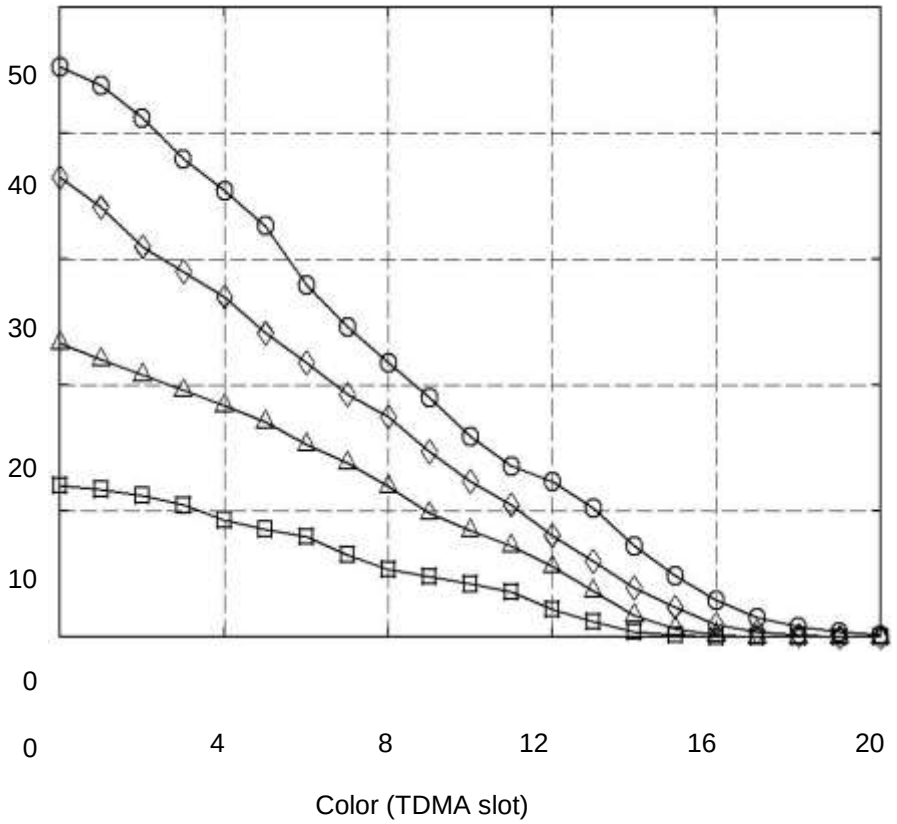


Figure 2.6: Number of transmission nodes assigned to each color when 8 cycles and 21 colors are used.The network size is 100, 200, 300 and 400, respectively (from bottom to top).

2.4 Effects of nodal mobility

2.4.1 Nodal mobility

Nodal mobility affects the FPRP protocol in two ways. One is on the opera-tion of the protocol itself. The other is on the transmission schedules generated by the protocol. From the view point of a protocol, to be robust to mobility requires that either it has some redundancy to function correctly in the pres-ence of topology change, or its operation is not signi cantly affected by topology change without taking special precautions. The duration through which node movement can have a negative effect on a protocol can be called its \suscepti-ble window". The longer the susceptible window, the less robust the protocol. For a protocol which requires collecting the

entire network topology ([21, 23] for example), the susceptible window is very long and the protocol is fragile. The ve phase dialogue is designed without explicitly taking node movement into consideration. It is relatively robust because it has a very short susceptible window. If a node moves into a neighborhood in the middle of a ve phase reservation cycle, it could miss the opportunity of detecting a collision (note that collision detection is accomplished in the rst two phases) and su ers a collision in the corresponding slot. However, the effect of node movement in a reservation cycle does not propagate or accumulate. In a reservation cycle, the topology in the neighborhood is explored *at the same time* as the schedule is produced, and this topology information is not passed onto the next cycle. The topology information explored in a cycle is always up-to-date, and no obsolete information is involved. Consecutive reservation cycles are independent in terms of collision detection, and consequently the susceptible time of the FPRP proto-col is simply the length of a reservation cycle and is therefore very short. Most nodes could hardly move in the duration of a reservation cycle and the network is largely static. For the same reason, the pseudo-Bayesian estimation of the parameters n_b; n_c is not signi cantly a ected by the nodal mobility either. We conjecture that the FPRP protocol is among the most robust protocols generat-ing topology-dependent transmission schedules, because other protocols require topology and/or schedule information, either for the entire network or in a neigh-borhood, collected *before* the schedule of a node can be computed. Therefore their susceptible windows are much longer and they are less immune to nodal mobility.

Once the topology-dependent transmission schedule is calculated, before it is updated or regenerated, it is subject to corruption caused by topology change. This process can be called \aging". The robustness of a schedule can be measured by $P_{crpt}(;t)$, the probability that a transmission scheduled in a slot is corrupted under mobility after a certain length of time t. Because every transmission is a broadcast, a transmission is considered corrupted if any of the one-hop neigh-bors of the transmitter cannot receive its packet correctly. Typically $P_{crpt}(\ ; t)$ increases with the nodal mobility ( ) and the observation time (t). If t is the length of an information epoch, the schedule is regenerated every t seconds and $P_{crpt}(\ ; t)$ is the probability that a transmission is corrupted before it is resched-uled. $P_{crpt}(\ ; 0)$ is the probability that a

scheduled transmission is corrupted immediately after the FPRP protocol completes its operation, and is therefore a measurement of the robustness of the FPRP protocol itself. It is clear that the more frequent the schedule gets updated, the less effect the nodal mobility has. This is a compromise between reducing the corruption probability and reducing the scheduling overhead. Because in large network the FPRP protocol produces the broadcast schedule very quickly, it can be executed more often than other, time-consuming scheduling protocols, such as [19, 22, 21, 23, 25]. Later we will see that with the FPRP protocol, it is possible to maintain a low corruption probability while still keeping a low scheduling overhead.

We would like to point out here that the \aging" process of a transmission schedule is a property of the schedule itself, and every schedule produced by a greedy algorithm of some form, such as the algorithms in [19, 21, 22, 23, 25], is equally susceptible to node movement. This is because in the slot assignment, robustness is sacrificed for bandwidth e ciency. Although not explored here, it might be bene cial to balance the number of transmission nodes in di erent slots, since the rst few slots are over utilized with greedy algorithms, therefore more prone to corruption, than the latter slots (Figure 2.5).

2.4.2 Simulation results

The simulated network has 100 nodes (N = 100), where initially every node is placed randomly in a closed area of 10 by 10. The transmission range of R = 1:5 unit length is the same for all of nodes. We assume the transmission range is 1 km and the transmission rate is 1 Mb/s. The nodes move randomly, and when a node moves and hits the boundary, it is bounced back.

The simulations are performed with two di erent mobility models, one is a Brownian motion model (BM) and the other is a randomized constant speed movement model (RCS). Under the Brownian motion model, every node per-forms independent random walk in both X and Y directions with step size of h every seconds. The combined effect is that every seconds, a node randomly chooses one of four possible directions (NE,NW,SE,SW) with equal probability and makes a move of size $2h$. The speed S of this movement is ^{2h}. A possible scenario is a large tank battalion with hundreds of tanks moving in the same general direction. The relative motion between the tanks is Brownian motion-like. Because the minimal time unit in the FPRP protocol

is a phase, we take as the duration of a phase. We estimate = 40 s, including transmission time, propagation time and the time for the transceiver to switch between transmis-sion and receiving mode. With the movement pattern and the time unit ( ) fixed, the nodal mobility can be determined solely by the node speed S. Un-der the randomized constant speed movement model (RCS), every node moves with constant speed S in a randomly picked direction. Once the direction is determined, the node movement is deterministic. This is similar to the \ran-dom waypoint model" in [37] with a pause time 0. Apparently this model has a more severe effect on the algorithm performance than the Brownian motion model. This model simulates a group of autonomous vehicles moving in a large working area. For both the BM and the RCS model, the degree of mobility can characterized by speed S. Hence we adopt the notion = S and use the two interchangeably. Simulations are performed for 1000 times under both models and the results are averaged. Figure 2.7 shows the results under the BM model and Figure 2.8 shows the results under the RCS model.

From the simulations we can see that the FPRP protocol itself is very robust under a wide range of mobility, regardless of the mobility model used. This can be seen from $P_{crpt}(; 0)$, which is the probability of a slot being corrupted immediately after the schedule is generated. Under both mobility models (Fig-ures 2.7 and 2.8), $P_{crpt}(; 0)$ is relatively insensitive to node movement. Also, the scheduling e ciency, measured by the number of slots assigned, and the scheduling overhead, measured by the number of reservation cycles used, remain largely unchanged when the network becomes more volatile. The average num-ber of slots assigned is 16 and the average number of reservations cycles is 89. This is due to the fact that the protocol has a susceptible window of only 200 s (the length of a ve phase reservation cycle), and the entire scheduling process is complete in 89 cycles. Even the rst reserved slot, which is the worst case in terms of aging, has only \aged" for 18 ms (the length of 89 reservation cycles) by the time the scheduling operation ends. If the number of reservation cycles is preassigned and fixed, as discussed earlier, we estimate 150 reservation cycles, or 30 ms, will be enough. It is clear that network is largely static during 30 ms, and nodal mobility does not have a signi cant effect on the FPRP protocol itself. The protocol can be executed frequently, for example once every 1 second, to maintain the

transmission schedule fresh enough, without incurring too much overhead (3% of the total bandwidth).

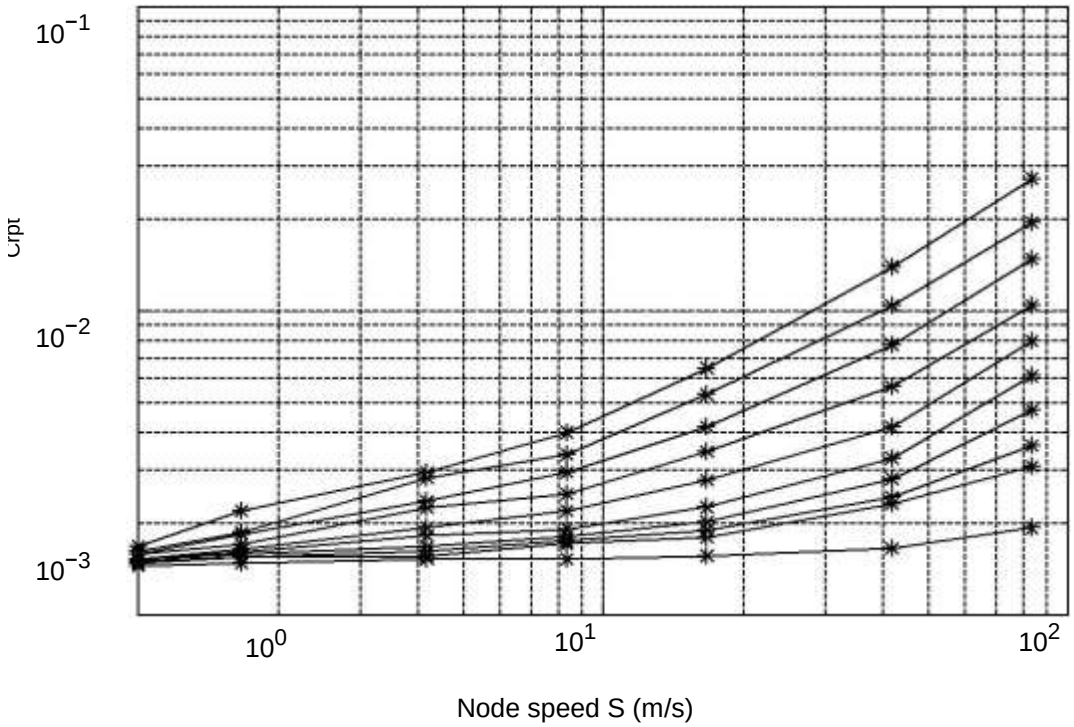


Figure 2.7: Slot corruption probability $P_{crpt}(S; t)$ under the Brownian motion model (BM). The observation time t is 0, 0.5, 1, 2, 4, 8, 16, 32, 64, 128 seconds respectively (from bottom to top).

Unlike the FPRP protocol itself, the transmission schedule depends heavily on the mobility model. A transmission is more likely to be corrupted when the nodes move in randomized constant speed motion than with Brownian motion. When the nodes move with a speed 10 m/s, $P_{crpt}(S; t)$ is approximately 0.02 and 0.03, after 0.5 and 1 second respectively, under the RCS model, as opposed to less than 0.002 under the BM model. When the observation time t increases, the corruption probability increases more quickly under the RCS model, and very soon it becomes unacceptably high. Frequent scheduling is more important in this case. The RCS model represents the worst case and forces the schedule to be updated at a rate of once of every second. This way the collision probability can be kept su ciently low for the envisioned network.

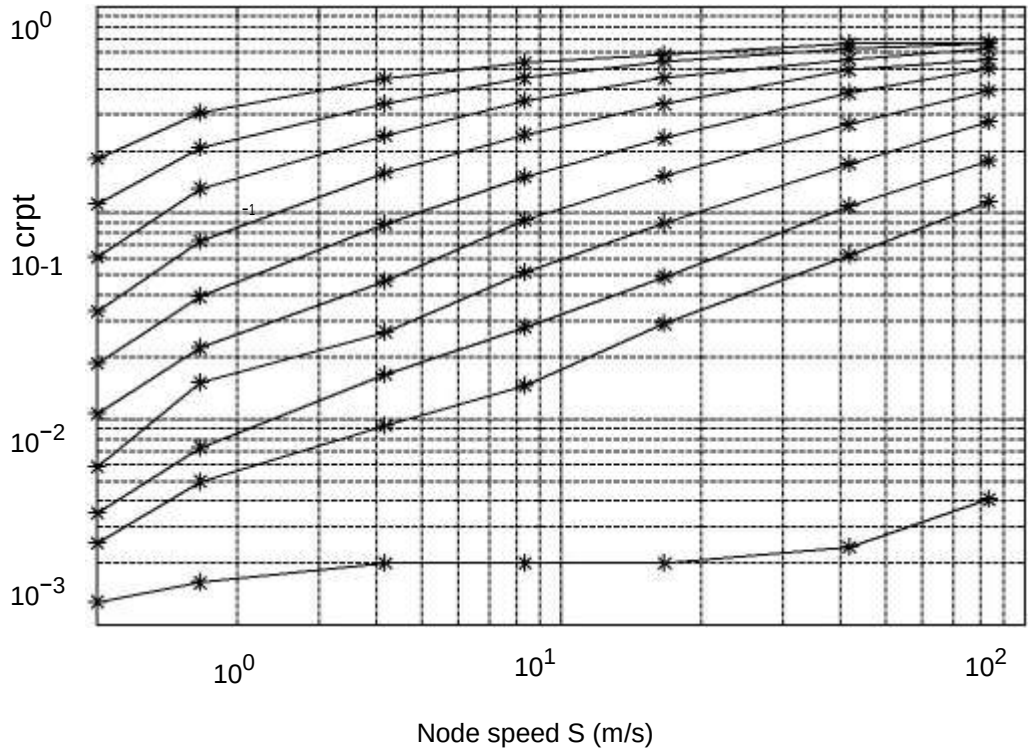


Figure 2.8: Slot corruption probability $P_{crpt}(S;\ t)$ under the randomized constant speed movement model (RCS). The observation time t is 0, 0.5, 1, 2, 4, 8, 16, 32, 64, 128 seconds respectively (from bottom to top).

2.5 Applications

So far no reason has been given as to *why* the nodes contend for the slots. This depends on the nature of the network and its higher layer protocols. The FPRP only provides a *means* for the nodes to make TDMA broadcast slot reservations. Nodes can make their reservations depending on their tra c load. The TDMA schedules produced thereof can be used to transmit user generated packets. The FPRP can also be used to make broadcast reservations for network control tra c. A broadcast schedule is very useful when the network control/reorganization is performed. Since a node can reserve a TDMA slot and participate in a network-wide organization/control phase, the FPRP is well-suited for supporting dis-tributed network control protocols. It is particularly useful as an initial signaling channel in an ad hoc network, where nodes need to explore their neighborhood and exchange connectivity and control information.

The fact that a node needs no a prior knowledge about the network makes it ideal for such a \rendezvous" role. Even nodes from di erent networks can merge together with the help of the FPRP protocol.

As we have seen, it is feasible to use the FPRP protocol to regenerate the transmission schedule frequently. Each time the protocol is executed, the existing schedule is discarded and a new one is generated from scratch. In fact, this does not have to the only solution. In some cases, it is possible to update the broadcast schedule gradually, i.e. only the schedule for the moving portion is modified. In the next chapter, we will use FPRP as the basis of another distributed TDMA slot reservation protocol (E-TDMA). In E-TDMA, the nodes of a mobile ad hoc network use the FPRP to obtain one or more slots in the control frames for making reservations for information slots, thus allowing the nodes to organize themselves autonomously.

2.6 Conclusion

A new TDMA slot assignment protocol, viz. the FPRP, has been presented. It allows nodes in a mobile ad hoc network to reserve TDMA broadcast slots and form broadcast schedules. It jointly and simultaneously performs the functions channel access and graph coloring. It does so without any centralized mechanism or constraint on scalability. It requires minimal computation capability in the nodes and can be easily implemented, provided a time synchronization signal of su cient accuracy is available, and a node is able to distinguish between the case of one and multiple packets arrivals. It works best in a network where nodes are uniform and form a mesh-shaped topology, and the nodal degree can be well estimated and built into the protocol. Simulation results showed that it can generate transmission schedules with good quality with a reasonably low amount of overhead, and is not a ected much either by the network size or by nodal mobility. Therefore it is well-suited for use in large, mobile ad hoc networks.

Chapter 3

An Evolutionary-TDMA scheduling protocol (E-TDMA) for mobile ad hoc networks

3.1 Introduction

In this chapter we develop a distributed protocol which generates and maintains TDMA transmission schedules which accommodate both a randomly chang-ing network topology and dynamic bandwidth requirement. In this protocol, termed Evolutionary-TDMA scheduling protocol (E-TDMA), a node can reserve conflict-free time slots for transmission to one (unicast), or some (multicast), or all (broadcast) of its one-hop neighbors. The resulting schedule is a mixture of unicast, multicast and broadcast transmissions. The protocol deals with the frequent changes in the network topology and in the tra c pattern by frequently updating the current schedules in an incremental, or evolutionary manner. The schedules can be updated at many parts of the network simultaneously, and a node only interacts with its neighbors for reserving time slots. The operation of the protocol is not affected by the network size but only by the node density, thus it is a scalable protocol.

The organization of this chapter is as follows: we rst discuss some consid-erations of TDMA scheduling protocol in general and outline what we believe is important. To a large extend the design of the E-TDMA protocol is guided by these considerations. We then describe the protocol itself and prove some important properties. Pseudo-code of the protocol is given in the Appendix. After illustrating its operation with an example, we present simulation results of the protocol and compare it with the standard IEEE 802.11 protocol.

3.2 Design Considerations for TDMA Trans-mission scheduling

The radio channel readily supports broadcast communications. When a node transmits using an omni-directional antenna, its packet reaches every node within its transmission range. A transmission is successful if the packet is the only one reaching the receiver, and the receiver

itself is not transmitting at the same time. With TDMA, a node cannot transmit and receive simultaneously (no pri-mary interference), and it cannot receive more than one packets at a time (no secondary interference). If we do not consider the capture e ect, from a trans-mitter's point of view, when it is transmitting a packet to an one-hop neighbor, it is blocking all the other neighbors from receiving from other sources; from the receiver's point of view, to receive a packet successfully prohibits all its one-hop neighbors, except the intended transmitter, from transmitting. Scheduling in a multihop network like this can be tricky, because nodes as far as two hops apart can conflict, but cannot communicate directly with each other (they are said to be \hidden" from each other). A transmission can be classified depending on the number of its designated receivers: unicast, multicast or broadcast, designating delivery to one, or some, or all of the one-hop neighbors of the transmitter, re-spectively. Multicast transmission can be viewed as the general case with an arbitrary subset of one-hop neighbors as receivers, while unicast and broadcast are the extremes with one or with all the neighbors as receivers. The transmis-sion requirement found in a real ad hoc network is often a mixture of unicast, multicast and broadcast, where the majority of the data tra c will likely be uni-cast and multicast|with broadcast typically being used for network control and management activities. The amount of bandwidth required by di erent nodes can vary dramatically. A node should be able to reserve di erent amounts of bandwidth, possibly using di erent transmission types.

In the parlance of graph theory, transmission scheduling in an ad hoc net-work is equivalent to a graph coloring problem, with each transmission slot represented by a distinctive color. Generation of a unicast schedule is equivalent to edge coloring, whereas generation of a broadcast schedule is equivalent to node coloring. Generation of a multicast schedule is to color multiple edges| each connected to a same node (the transmitter). Scheduling all three types of tra c is a mixture of node coloring and edge coloring. The coloring constraints are the same as the requirements for conflict-free transmissions. To produce the optimal schedule (where optimality is measured in terms of bandwidth ef-ciency; i.e. we desire schedules with the minimum number of TDMA slots) is NP-complete [38, 22, 27]. To nd the maximum transmission set (the set of nodes that can transmit simultaneously without mutual interference), either directly or incrementally, is also NP-complete [39, 40]. However, for a mobile network, the most

bandwidth-e cient schedule might not be the best. It has the highest spatial reuse and the least redundancy. Therefore it is very fragile and is susceptible to being corrupted. When nodes move, the topology of the network changes, and collisions may occur in the schedules, even though these schedules were conflict-free when they were generated. The schedules also need to accommodate changes in the bandwidth requirements. As old transmission sessions end and new sessions begin, bandwidth should be released from termi-nated sessions and assigned to new sessions quickly. All these changes, both in network topology and in network tra c, require the transmission schedule to be updated frequently. This is referred to as schedule \maintenance". Because maintenance needs to be done very often, it has to be cost-effcient. Compared with other types of networks, an ad hoc network is limited both in bandwidth and in computation power. It is desirable that the communication and computa-tion overheads required to generate and to maintain the transmission schedules be as low as possible. A brute force approach, which tears down the existing schedules completely whenever changes occur in the network and regenerates new ones, is apparently inappropriate. Although a new schedule reflects the latest network topology and bandwidth requirements and can be made very ef-cient, its generation is likely too costly and somewhat redundant, especially when only a small part of the existing schedules is outdated and the rest is still valid. A more natural solution is an incremental, or evolutionary approach. In such an approach, the existing schedules are kept as much as possible. Only the part which is outdated, either due to node mobility or due to changed bandwidth requirements, is changed. If the interval between two updates is short enough, only a small portion of the existing schedule needs to be changed. Compared with regenerating the entire schedules, this method is more economical. Incre-mental scheduling protocols have been studied in [41, 42, 28, 40]. Due to the dynamic nature of an ad hoc network, distributed protocols are preferred. This is important both for e ciency purposes and for robustness. It is desirable that the scheduling process does not depend on a particular node. A real network could be extremely dynamic, both in size and in topology. The nodal density could vary dramatically as nodes get together or disperse in a large area. The network could be partitioned, and when partitioning occurs each portion should operate by itself as a smaller network. This requires the protocol to be scalable, i.e. it can perform equally well in a large network as in a small network.

In order to generate or update the transmission schedules quickly, one should take advantage of the local nature of the transmissions. Transmission from a node only reaches its one-hop neighbors and affects nodes up to two hops away. In order to make the schedule conflict-free, it is su cient for a node to know only those transmissions in its two hop range. Nodes far apart from each other can schedule their transmissions independently. This makes it possible to de-sign protocols generating the schedules on a local basis. Recently a class of hybrid protocols which combine contention and reservation have been proposed [43, 44, 45, 46, 47]. These protocols use contention for making reservations, thus eliminate the need for the nodes to wait in turn to reserve their slots. Because contention only involves nodes nearby, these protocols are scalable. It is useful when the network is large and the schedule needs to be updated frequently. Un-der these demanding requirements it is more important to generate a conflict-free schedule quickly than to spend the time to generate a highly e cient schedule.

The preceding highlights what we consider to be important characteristics for a scheduling protocol. Here our intention is not to produce the most bandwidth-e cient schedule, but to produce and to maintain a conflict-free schedule as rapidly as possible in a fully-distributed, parallel fashion with only local knowl-edge. The design of the protocol incorporates almost all of these characteristics, falling short principally in the ability to handle large variations of nodal density. The result is the E-TDMA protocol.

3.3 The Evolutionary-TDMA Scheduling Pro-tocol

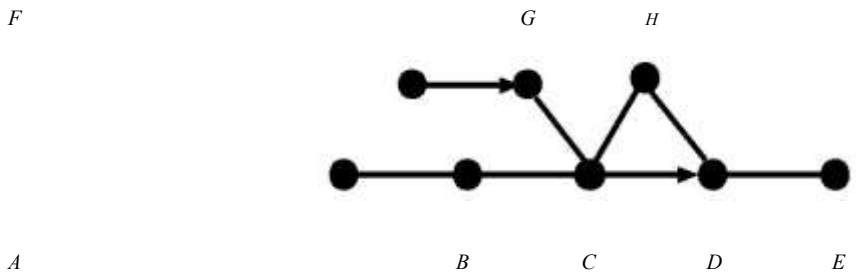
The E-TDMA protocol allows nodes to assign TDMA transmission slots among themselves as network composition and bandwidth demands change. The proto-col produces two TDMA schedules simultaneously, each used in di erent portion of the same channel and for di erent purpose. The rst schedule is a broadcast schedule, in which every node is assigned one slot. This broadcast schedule is used for nodes to exchange information in the control frame and is called the con-trol schedule (*ctrl schedule*). The second schedule carries user generated tra c in the information frame, and is called the information schedule (*inf o schedule*). All reservations here are *one-hop* reservations. In the *inf o schedule*, a node can reserve di erent amount of bandwidth to transmit to one (unicast), or some (multicast), or all (broadcast) of its neighbors,

depending on its need. Both the *ctrl schedule* and the *inf o schedule* reflect the topology of the network. Fur-thermore, as the network topology and the bandwidth requirements change, the schedules adjust accordingly to maintain conflict-free transmissions. The algorithm copes with changes in the network topology and bandwidth requirements in an incremental manner in order to minimize the re-scheduling overhead and to support QoS to the extent possible in these networks.

With the E-TDMA protocol, all nodes participate in the scheduling process on an equal basis. The scheduling process is executed across the entire network at the same time. Nodes do not wait in some particular order to schedule their transmissions. They determine who can reserve transmission slots by contending for a permission (called a temporary color), and many nodes can acquire this permission and schedule their transmissions simultaneously. This reduces the overhead and enhances the robustness. Essentially, every node is responsible for its own transmission schedule. A node can reserves a conflict-free time slot to transmit to a set of its one-hop neighbors. If any of its receivers begin to su er a collision caused by another transmission due to some topological change, the transmitter learns this from that receiver and stops transmission in the slot. It can reserve another time slot if it needs to. After a transmission is complete, the transmitter releases the slot, which can be reserved for another transmission. A node only needs to exchange information with its one-hop neighbors. Because of the local nature of the protocol, it is not sensitive to the network size. It is not affected by network partition either. It is suitable for a large, homogeneous network of changing size, such as a large, mobile military formation.

We make the following assumptions about the network:

Nodes keep perfect timing. Global time is available to every node and is tight enough to permit global slot synchronization; Every link is symmetric. The topology of the network can be represented by an undirected graph;



3. *(state(s) = Idle)*
4. *(state(s) = Block_r)*
5. *(state(s) = Trans, target(s) = D)*
6. *(state(s) = Recv, target(s) = C)*
5. *(state(s) = Block_t)*
6. *(state(s) = Trans, target(s) = G)*
7. *(state(s) = Collision)*
8. *(state(H) = Block_tr)*

Figure 3.1: Slot states de ned by the E-TDMA protocol.

The network topology changes slowly relative to packet transmission time; Every node is able to operate the Five Phase Reservation Protocol (FPRP); Packet collision is the only source of receiving error.

3.3.1 Notations used by E-TDMA

With E-TDMA, the activity of a node n_i in a given slot s is represented as a pair ($state(s)$; $target(s)$), where $state$ is the activity of this node in slot s, and $target$ is a set of one-hop neighbors to which this node transmits to or receives from. Without causing confusion an one-hop neighbor is sometimes simply called a neighbor. With constraints required by conflict-free TDMA transmissions, the activity of a node n_i in a slot s can be classi ed into the following states:

Transmits to a set of neighbors R : ($state(s) = Trans$; $target(s) = R$). If the transmission is a broadcast, $target(s) = Broadcast$; Receives from a neighbor n_j : ($state(s) = Recv$; $target(s) = n_j$). For this case $jtarget(s)j=1$;

If a node is not transmitting or receiving in slot s, it is in one of the following states:

[34]. Blocked from transmitting because at least one of its neighbors receives from another node, and none of its neighbors transmits: ($state(s) = Block\ t$);

[35]. Blocked from receiving because at least one neighbor is transmitting to another node, and none of its neighbors receives: ($state(s) = Block\ r$);

[36]. Both blocked from transmitting because at least one neighbor is receiving, and blocked from receiving because at least another neighbor is transmit-ting: ($state(s) = Block\ tr$);

[37]. Experiencing a collision when it is supposed to receive from a neighbor: (*state*(*s*)= *Collision*);

[38]. Idle, when none of its neighbors transmits or receives in slot *s*: (*state*(*s*) = *Idle*).

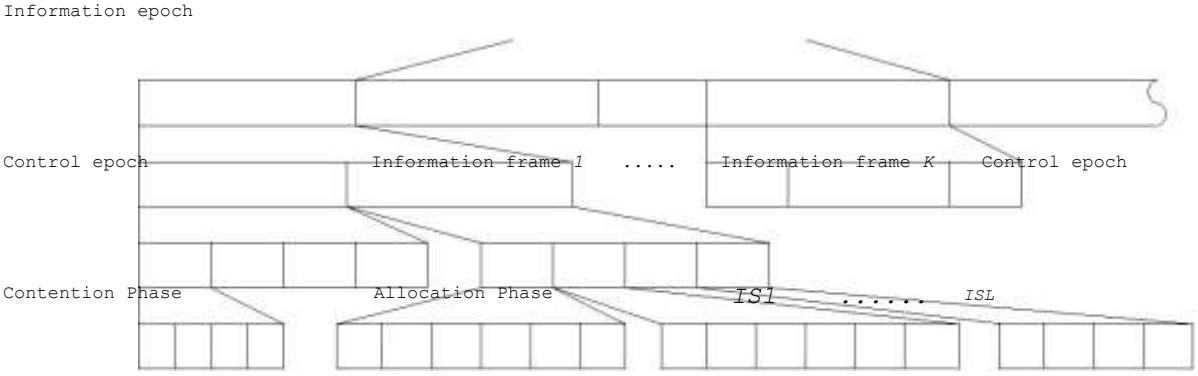
Note that the *target* eld is only defined for states *T rans* and *Recv*. For the other states *target* is not meaningful. These states are exclusive, i.e. a node is at one and only one of these states in any given time slot. Any slot when a node does not transmit can be called a passive slot. Figure 3.1 illustrates these di erent states. Suppose in a slot *s*, node *C* transmits to *D* and node *F* transmits to *G*, their transmissions to the intended receivers are shown with arrows.

Note that these transmissions also reach other one-hop neighbors of the trans-mitters due to the broadcast wireless medium (not shown). Node *D* can receive successfully from *C*, because *C* is its only neighbor transmitting in that slot. The transmission of *C* also reaches *G* and collides with the transmission from *F*. Node *G* su ers a collision in the slot. The states of other nodes (*A*,*B*,*E*,*H*) not in *T rans* or *Recv* are determined by their positions relative to the transmitters (*F; C*) and the intended receivers (*D; G*).

3.3.2 Frame structure of E-TDMA

The protocol operates within a single TDMA channel[y]. The channel is parti-tioned into two portions: control epoch where the schedules are updated by the protocol, and information epoch where user data transmission takes place. The two epochs are interleaved periodically. The frame structure of the protocol is de ned in Figure 3.2. An information epoch has *K* number of information frames, which in turn is consisted of *L* number of information slots. In an information slot, a node transmits or receives a data packet (or a fragment of a data packet) with its neighbors according to the *inf o schedule*. How many slots a node needs in the *inf o schedule* and which neighbor(s) the transmission in a slot is addressed to depends on the type and the amount of out-going tra c at this node and can be time-varying. E-TDMA accommodates these transmission requirements by updating the *inf o schedule* periodically. The *inf o schedule* is updated in the preceding control epoch. A control epoch has two phases: a contention (*C*) phase

and an allocation (*A*) phase. A contention phase is divided into *N* contention slots, each of which is consisted of a number of Five Phase Reservation Protocol (FPRP) cycles. A contention slot corresponds to a tempo-rary color (de ned later), and if a node needs to acquire a temporary color, it contends with the FPRP protocol in the corresponding *C* slots. If successful, it reserves the temporary color for the current control epoch. An allocation phase has *N* number of frames. In an *A* frame, nodes exchange information with their one-hop neighbors by transmitting according to the *ctrl schedule*. A transmis-sion in the *ctrl schedule* is a broadcast, hence the *ctrl schedule* is a broadcast schedule. In the parlance of graph theory, it corresponds to a distance-2 node coloring. For this reason a slot in the *ctrl schedule* is also called a color. There are two types of colors in the *ctrl schedule*: *N* temporary colors and *M* per-manent colors. A node has at most one permanent color and one temporary color in the *ctrl schedule*. A temporary color is a permission to reserve new information slots or permanent colors. If a node needs to make new reservation in a control epoch, it rst needs to acquire one of these permissions. Its tem-porary color becomes invalid after this control epoch, and if it wants to make another reservation later it has to contend again. The permanent color of a node lasts much longer. A node needs a permanent color in the *ctrl schedule* for ex-changing its scheduling information with its neighbors (but not for making new reservations). Once a node acquires a permanent color, it transmits in every slot designated this color as long as its transmission does not collide with others. If a collision occurs due to some topological change, a node will discard its current permanent color and reserve a new one. How a node acquires its permanent color will be described later. Di erent *A* frames have di erent lengths. The



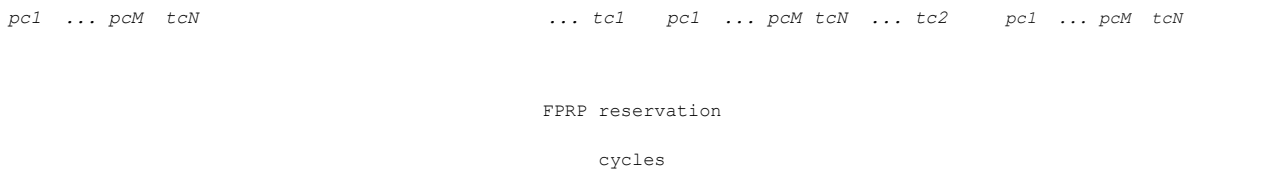


Figure 3.2: Frame structure of E-TDMA. There are M permanent colors ($pc1; :::; pcM$) and N temporary colors ($tc1; :::; tcN$). There are K information frames in an information epoch and L information slots ($IS1; :::; ISL$) in an information frame.

First A frame ($A1$) has M slots corresponds to the M permanent colors and N slots corresponds to the N temporary colors. Slots corresponds to the temporary colors are placed after the permanent colors and are arranged in reverse order. The number of slots corresponding to the temporary colors decrements in each following A frame. Temporary color $tc1$ only appears in $A1$, $tc2$ only in $A1$ and $A2$, and so on. The last A frame has only $M + 1$ slots corresponding to the M permanent colors and temporary color tcN.

3.3.3 Details of E-TDMA

In E-TDMA, every node generates and maintains its own schedules in collab-oration with its neighbors. The schedule of the entire network is simply the collection of the schedules of all the nodes. No single node has global informa-tion such as the size, the membership or the schedules of the entire network. A node only interacts with its one-hop neighbors directly. For a given node, an other node two hops away could cause interference if they both transmit in the same slot, but such interference or a collision takes place at an one-hop neighbor. By knowing the schedules of this one-hop neighbor, this node learns the relevant part of the schedules of this two-hop neighbor indirectly. This way it is able to avoid the collision. For example, suppose nodes $n1$ and $n2$ are one-hop and two-hop neighbors of a node $n0$ respectively. If $n2$ reserves a slot to transmit to $n1$, by knowing that $n1$ receives from $n2$ in that slot, $n0$ learns the transmission of $n2$. It will avoid transmission in the same slot in order not to interfere. This way the transmission information of the two-hop neighbors is embedded in the schedules of the one-hop neighbors, and nodes two hops apart do not need to communicate directly. A node does not need or have any information about nodes beyond its two hop range. Nodes exchange their schedules periodically in the control epochs. A node keeps its neighbor information in a list $N B$.

By tracking its neighbors and their schedules, it knows which slots are in use and which slots are available. It uses this information to reserve its new slots. Many nodes can reserve their transmission slots simultaneously. Because this set of nodes that reserve their slots at the same time is determined by the FPRP pro-tocol, they are likely to be three or more hops apart from each other. They can reserve their time slots independently without causing collisions.

When the schedules are updated, they are always updated on the basis of the existing schedules. A reservation is only released when its transmission is complete, or when it su ers a collision, but never released to accommodate a new reservation. A new reservation can be made only if it does not conflict with any reservations established earlier. The resulting schedules evolve over time with the changing topology and tra c pattern. This gives E-TDMA protocol its name. In order to illustrate the incremental nature of the protocol, we describe below how it works in a scenario where a set of existing schedules (*ctrl schedule* and *inf o schedule*) have already been running in the network. It will be seen later that network initialization, where the old schedules are non-existent, is a trivial extension of this scenario.

The Contention (*C*) Phase

The purpose of the contention phase is to assign the *N* temporary colors to the nodes. Remember they are permissions for reserving new slots in a control epoch. If a node needs to reserve a permanent color for its *ctrl schedule* (if it does not already have one), or an information slot in the *inf o schedule* (if it has new tra c arrival and requires more transmission bandwidth), it rst contends in the *C* phase for a temporary color with the FPRP protocol. The FPRP protocol ensures that only nodes three hops apart or further can acquire the same temporary color ^{z}. If successful, a node will transmit in slots corresponding to this temporary color (along with any slots corresponding to its permanent color) in the allocation phase of this control epoch. In one of these slots it will reserve the times slots it needs.

The Allocation Phase

In this phase a node transmits its current schedules in a schedule update packet (*su packet*) in a slot designated to its permanent or temporary color, and listens for schedules transmitted by its

neighbors in other slots. It updates its list NB as it receives transmissions from others. If the entry of a neighbor in NB has not been updated for some time, that neighbor is deemed to have moved away (or powered o) and its entry is deleted. As a node listens for the schedules of the others, it also makes adjustments to its own schedules based on what it hears. Among the possible states of a slot described early, the most important states are $Trans$ and $Recv$. We describe under what conditions a node can start (or stop) to transmit to (or receive from) an one-hop neighbor.

For a node to stop transmission in a slot is to release a slot it has previ-ously reserved but no longer needs. A node may release an information slot in $info\ schedule$ when its transmission to the receiver(s) has completed, or when learning its transmission is colliding at a receiver (by hearing the schedule broad-casted by that receiver), or release a slot in the $ctrl\ schedule$ (a permanent color) after it nds its transmission in this slot is having a collision at one of its one-hop neighbors. Unused slots are released in the beginning of the allocation phase. To release a slot s, a node simply changes its state of the slot from $Trans$ to one of the passive states, depending on the states of its neighbors. Its neighbors will be informed of this change when this node broadcasts its updated schedules. When a receiver of this transmission receives the broadcast, it learns the slot is released by the transmitter and stops receiving in that slot. A released slot can be reserved later for another transmission.

To reserve a new slot requires more care than to release a slot, due to the pos-sible conflict caused by this reservation. In the ith allocation frame, only nodes which acquired the ith temporary color in the preceding contention phase can reserve new information slots and permanent colors. There can be many nodes in this set, and the size of this set is likely to grow with the size of the network.

In the ith allocation frame, the slot designated to temporary color tci is located at the very end. A node with color tci chooses its new transmission slots just before it announces its schedules to the neighbors in this last slot. (If a node also broadcasts in an early slot of this frame designated to its permanent color, it is not allowed to choose its new slots then.) By this time it has received

broadcasts from all its one-hop neighbors with valid permanent or temporary colors and has learned their schedules. This node now chooses its new transmission slots based on these latest information. It can reserve a permanent color in the *ctrl schedule* if it needs one, and reserve information slots in the *inf o schedule* depending on its tra c requirement. If node n_i wants to reserve a new slots transmitting to a neighbor n_j , it picks a slot s when the receiver n_j is either *Idle* or *Block t*, and itself is either *Idle* or *Block r*. If there are multiple slots satisfying the criteria, a node chooses one of them randomly [x]. A node incorporates its new reservations into its schedules and broadcasts the updated schedules to its neighbors. The receivers of its new transmissions changes their states in the corresponding slots to *Recv*. A reservation is established this way. Transmission takes place in this reserved slot in the following information frames until the transmitter releases the slot.

After the *i*th allocation frame, every node with temporary color *tci* has had a chance to reserve new information slots and permanent colors. Whether they are able to reserve the slots they need depends on the current load and the schedules of its neighbors. The temporary color *tci* is no longer useful, so it disappears from the rest of this allocation phase.

The Appendix contains a pseudo-code of E-TDMA which provides more de-tails. The schedules generated by E-TDMA satisfy the following conditions given as Theorems 3.1 and 3.2.

Theorem 3.1: If nodes in a network do not move, the schedules produced by E-TDMA are conflict free for every node with a valid permanent color.

In order to prove Theorem 3.1 we need the following two lemmas.

Lemma 3.1: If a node n_i knows the up-to-date schedules of all its one-hop neighbors, it is able to pick a slot (a permanent color or an information slot) to transmit to one (or some, or all) of them. This transmission will not collide with any on-going transmissions.

Proof: When a node n_i wants to find a slot to transmit to a set of receivers R $N B$, if it knows the current schedules of all its one-hop neighbors, it can pick a slot s when its own state is either *Idle* or *Block r*, and the state of every node n_j *2 R* is either *Idle* or *Block t*. The receivers are

able to receive successfully, because for them there is no other 1-hop neighbors transmitting in the same slot (otherwise they would be either *Recv* or *Block r* or *Block tr*). The transmissionwill also reach other one-hop neighbors of the transmitter (*N B \ R*), but it will not cause interference to them. If such interference would occur, at least oneof these nodes, n_j *2 N B \ R*, is receiving from another source (*Recv*), and the state of n_i would be *Block t* (or *Block tr*). But the state of n_i is either *Idle* or *Block r*, therefore a conflict. Q.E.D.

Lemma 3.2: If two nodes are at least three hops away, their transmissions will not interfere with each other.

Proof: This is apparent since their transmissions only reach their respective one-hop neighbors. Because they do not share any common one-hop neighbor, their transmissions will not interfere. Q.E.D.

We proceed to prove Theorem 3.1:

Proof of Theorem 3.1: Because new transmission slots are reserved only in the allocation frames, we focus on these frames. In frame *Ai*, only nodes with temporary color *tci* are allowed to reserve new permanent colors and infor-mation slots. Other nodes can release slots they previously reserved, but such releases will not cause any conflict in the schedules. A node broadcasts its cur-rent *ctrl schedule* and *inf o schedule* to its neighbors in the slot designated its temporary or permanent color. By listening for its transmission, its one-hop neighbors learn its schedules. The slot corresponding to *tci* is located at the end of *Ai*, thus by the time a node with temporary color *tci* is ready to choose its permanent color or information slots, it has received broadcasts from all of its one-hop neighbors with valid permanent (and temporary) colors. The schedules in its *N B* list are up-to-date, and these schedules do not change at this time. From Lemma 3.1, any permanent color or information slot it chooses does not collide with any previously established reservations. The set of nodes with tem-porary color *tci* is determined by the FPRP protocol, and these nodes are at least three hops apart. From Lemma 3.2, no matter what permanent colors or information slots they choose, no collision will take place among them. Therefore if the schedules are conflict-free at the beginning of a control epoch, it remains

conflict-free after the control epoch. When the network is rst turned on, no slots are reserved, and the schedules are conflict-free. By induction the schedules are always conflict-free. Q.E.D.

When nodes start to move, collision may occur in their schedules, even if these schedules were conflict-free when they were generated. However, with E-TDMA, these collisions will not last long. Theorem 3.2: When nodes move, collision could take place in the schedules. For a node with a valid permanent color, a collision could last at most for the duration of an information epoch.

Proof: For two transmissions (n_i ! n_j) and (n_k ! n_l) in a same slot s, collision takes place at node n_l if n_i moves close to n_l and becomes its one-hop neighbor. Now two packets, one from n_i and one from n_k , reach n_l in slot s, and node n_l starts to experience a collision. When this happens, n_l changes its state of s to *Collision*. When n_l broadcasts in the next control epoch in a slot designated to its permanent color, both n_i and n_k receive the latest schedules from n_l. (For n_i, n_l is a new neighbor and is added to its list $N\ B$.) When these two nodes update their own schedules based on this newly received information, n_k stops transmission because it target n_l is having a collision. Node n_i may keep transmitting to n_j in slot s if n_j does not have a collision in s. This way the conflict at n_l is resolved at the next control epoch, and the longest time a collision lasts is the duration of an information epoch. Q.E.D.

When a node loses a slot due to collision, it tries to reserve another slot in the next control epoch. To reduce the duration of the collisions, it is desirable to update the schedules more often and thus to have short information epochs. However, unless the length of the control epoch is reduced accordingly, this in-creases the overhead of E-TDMA. As a compromise, one can choose the shortest control epoch, namely to let the number of temporary colors be 1, while reducing the length of the information epochs (by reducing the number of information frames) until the overhead of E-TDMA reaches its maximal allowance. This is the approach we take in our implementations in the simulator. However, the frame structure described early, where the number of the temporary colors is a design variable, is still useful if the length of the information epoch is determined by some other considerations. When a node is turned on, if there are other

nearby nodes already operating, it can learn the schedules of these neighbors and build up its $N\ B$ list by listening for their broadcasts. After it acquires a permanent color, this node becomes fully operational. If every node is turned on at the same time, both the $ctrl\ schedule$ and the $inf\ o\ schedule$ are null everywhere in the network. All the nodes would contend in the beginning because they all need permanent colors. In each control epoch, some nodes would succeed, rst to acquire a temporary color then to acquire a permanent color, and become operational afterwards. There is no di erence from the protocol's point of view, therefore E-TDMA does not have an explicit \network initialization phase".

Note that the schedules are conflict-free only for nodes with valid permanent colors. For a node n_i without a permanent color, it cannot make its current schedules known to its neighbors. Therefore there is no guarantee that packets will not collide at this node. The minimum number of permanent colors required to cover every node of a network is its distance-2 chromatic number, and is closely related to the maximal nodal degree (it is lower bounded by + 1). The number of permanent colors should be large enough that every node can acquire a permanent color with high probability. By providing only a fixed number of permanent colors, E-TDMA is limited by nodal density and cannot cope with the situation when all the nodes gather in a small area. It works best when the nodes are dispersed and the nodal density is uniform. A node never gives up its permanent color voluntarily. It loses its permanent color when the latter is corrupted by some topology change (i.e., another node with the same permanent color moves into its two-hop range). The lifetime of the permanent color of a node is therefore determined by nodal mobility. In a static network, a node keeps its permanent color forever. When the nodes move faster, the topology changes more frequently and a node loses its permanent color more often. It takes time for a node to regain a permanent color. For E-TDMA to work well, it is necessary that nodes do not move too fast. As a reservation-based protocol, E-TDMA fails when the network becomes too volatile. How frequently the schedules are updated determines how well E-TDMA handles network mobility.

By maintain the one-hop neighbor list $N\ B$, E-TDMA provides a neighbor discovery mechanism at the MAC layer. This eliminates the need for some routing protocols, such as AODV [11] and

TORA [13], to use their own neighbor discovery mechanisms. Although not used by E-TDMA itself, information about two-hop neighbors can be obtained as well (function get 2 hop neighbors in the pseudo-code). This may facilitates routing for some situations.

The way E-TDMA combines contention and reservation is unique among pro-tocols designed for ad hoc networks. In other protocols also using contention, like HRMA [44] or ADAPT [46, 47], nodes contend directly in the slots they want to reserve. They can only reserve unicast or broadcast, but not multicast transmissions. In E-TDMA, a node contends for a permission (a temporary color). With this permission a node can reserve multiple time slots. However, there are many similar protocols in other type of networks. In the D-TDMA protocol developed for cellular networks [48], a user terminal uses contention to make its bandwidth request known to the basestation. To transmit a bandwidth request packet to the basestation successfully is equivalent to acquire a tempo-rary color in E-TDMA. Both are permissions for reserving (potentially many) new time slots in a local area (a cell for D-TDMA and a two-hop neighborhood for E-TDMA). In D-TDMA, after a mobile terminal sends its contention packet successfully, the basestation assigns the new slots based on the request and the current schedules in the cell. A basestation is naturally a hub, and all the com-munications are between the basestation and the terminals. The basestation can easily manage the resources for all the nodes in its cell, because it has all the information. The scheduling is much easier because the network is only one hop. For a multihop ad hoc network, there is no natural centralized controller like a basestation, and nodes must negotiate with each other for making slot reservations. The multihop topology also makes scheduling more di cult. In E-TDMA, after acquiring a temporary color, a node has the sole right to reserve new slots in a two-hop neighborhood and it assigns time slots for itself. In fact if all the nodes of an ad hoc network are connected to one another and form a cluster, E-TDMA becomes D-TDMA without a basestation. The Markovian model developed for D-TDMA in [49] can be used to analyze the performance. But for E-TDMA, which is designed particularly to handle multihop topology, this is a rather uninteresting case and is in fact the worst possible scenario.

3.3.4 An example

We now illustrate via an example how E-TDMA updates the schedules (Fig-ure 3.3). There are 6 nodes (*A* to *F*) in the network, and the E-TDMA protocol has temporary colors (*tc*1*; tc*2) and 4 permanent colors (*pc*1 to *pc*4). There are 4 information slots in an information frame (*is*1 to *is*4). The original topology is shown in Figure 3.3.a. Suppose the control schedule and the information sched-ule were both conflict-free when they were generated according to the original topology, and these schedules are shown in Figure 3.3.c. Suppose node *E* moved towards node *C* and a new link appeared between them (Figure 3.3.b). This causes conflict in the original schedules, and the corrupted schedules are shown in Figure 3.3.d. Two transmissions, from *D* to *C* in *is*1 and from *F* to *E* in *is*2, are corrupted, and they need to reserve new time slots. We also assume that at the same time, node *A* needs to reserve a new slot to transmit to node *B*. So we will see how the protocol reallocates conflicting transmissions and accommo-dates a new one. When the next control epoch begins, the three nodes *A*, *D* and *F* , which require new information slots, contend for the temporary colors. Assume they all succeed, and nodes *A* and *D* acquire *tc*1 and node *F* acquires *tc*2. In *A*1, nodes *A* and *D* update their schedules after hearing broadcast from all their neighbors. Both of them schedule their transmissions in *is*4. The par-tially updated schedules after *A*1 are shown in Figure 3.3.e. In *A*2, node *F* with *tc*2 updates its schedule. It picks *is*3 for transmission to node *E*. The updated schedules after *A*2 are shown in Figure 3.3.f, where the conflicting transmissions are reallocated to new slots and the newly arrived transmission is also assigned a slot. Although only unicast transmissions are shown in the example, multicasts and broadcasts can be handled in similar ways.

3.4 Simulations

3.4.1 Implementation of E-TDMA

We have implemented the E-TDMA protocol with *N S* − 2 [50], a discrete event simulator widely used for network research. It is particularly popular in the ad hoc networking community, and many protocols used in ad hoc networks have been implemented, including IEEE 802.11, the standard wireless LAN MAC protocol, and a few routing protocols such as AODV, DSR and

TORA. This makes it easier to compare the protocols developed here with others. Without further explanation the parameters of E-TDMA used in the simulations are given in Table 3.1. higher layer protocol does not tell E-TDMA how much many slots to reserve, E-TDMA has to gure out the required bandwidth and reserve a corresponding number of slots. The unit for bandwidth in E-TDMA is an information slot, which is equal to

$$\text{slot} = \frac{\text{length of information epoch}}{\text{length of information epoch} + \text{length of control epoch}}$$

$$\#\text{ of bits transmitted in information slot} = \text{length of information frame}$$

(*bits=second*): (3.1)

Because an information slot is a large unit, the required bandwidth RB_i^b for transmission to an one-hop neighbor n_i is calculated in bps (bits per seconds) and converted to slots when needed. It can be calculated upon the arrival of a packet addressed to n_i with the following iterative algorithm:

$$RB_i^b = (1 -)RB_i^b + \quad L \quad ; \qquad (3.2)$$

Transmission rate	1 Mbps
Transmission range	250 m
# of permanent colors	12 (15)
# of temporary colors	1
# of FPRP cycles per temporary color	8
length of a FPRP cycle	200 s

# of frames per information epoch	4
# of slots per information frame	40
length of a *su* *packet* -	60 bytes
information bytes per slot	32 bytes
neighbor lifetime (*nb* *ttl*) -	3 control epochs
overhead per slot	4 bytes
slot guard time	20 *s*
bandwidth per information slot	18 kps

Table 3.1: Parameters of the E-TDMA protocol used in the simulations.

where L is the packet size in bits and T the last packet to n_i. In the beginning number of slots is given byis the time elapsed since the arrival of $RB_i^b = 0$ for all i. The corresponding

$$RB^s = ceil(RB^b = slot): \quad (3.3)$$

The parameter 0 < < 1 is used for smoothing the jitter of packet arrival. We use = 0:1 for the simulations. To prevent time slots from being locked forever after the last packet for n_i is transmitted (note Equation 3.2 updates bandwidth only when a packet for n_i arrives), RB_i^b and RB_i^s are reset to 0 when no packet arrives for n_i for sometime. We use 3 seconds in the simulation for this time-out period. A slot can also be released if explicitly required by the upper layer protocol. Because IP packets can have variable lengths, a packet often has to be transmitted in multiple time slots. Fragmentation and re-assembly will be needed in this case. Besides slots

required for user data transmission, every node also reserves a broadcast slot in the *inf o schedule*. This broadcast slot is used for transmission of control packets. Packets generated by routing protocols are often broadcast, and they are very irregular compared with user data packets. Without reserving a broadcast slot in advance, the delay for E-TDMA to reserve a slot upon the arrival of a control packet is unbearable. This broadcast slot is also used for user data packets when there is no control packet. Transmission of a packet may fail if it su ers a collision. A packet may also be dropped at the network layer if there is no route to the destination, or at the link layer if the interface queue is full (maximum length 50 packets).

The routing protocol used with E-TDMA is the QoS routing protocol de-veloped in Chapter 5. This protocol is based on Ad-hoc On-demand Distance Vector routing protocol (AODV) and can setup QoS routes for CBR tra c flows. It also generates best-e ort routes like the original AODV. We defer details of this QoS routing protocol to Chapter 5. E-TDMA does not work well with the original AODV protocol. The original AODV changes routes too frequently. Fre-quent route change requires frequent bandwidth reservations and puts a heavy burden on E-TDMA. With the QoS routing protocol, routes are more stable, and E-TDMA handles the bandwidth reservation required for QoS routes betterThe QoS routing protocol also reduces congestion by using multiple bandwidth-reserved routes. The amount of bandwidth used for packets transmitted on QoS routes are calculated by the QoS routing protocol, and the amount of bandwidth used for packets transmitted on best-e ort routes are calculated by E-TDMA. Because the time frames in E-TDMA are pseudo-periodic (the interleaved con-trol epoch make an information frame aperiodic), an information slot cannot synchronize with the data packets. Therefore we do not assume a source gen-erates one packet per frame or a packet is always transmitted in a single time slot. When there are multiple sessions transmitted to a neighbor and time slots are reserved for these sessions, packets from these sessions are multiplefixed and transmitted in all these slots. There is no one-to-one relationship between a time slot and a session

We compare E-TDMA with the IEEE 802.11 protocol. The 802.11 module in *N S* −2 was contributed by the MONARCH group at Carnegie Mellon University. The transmission rate of 802.11 is also 1 Mbps. With 802.11, bandwidth cannot be reserved as in E-TDMA, thus the QoS

routing protocol of Chapter 5 cannot be used. The original AODV is used with 802.11 in the simulations.

3.4.2 Simulation scenarios

A mobile ad hoc network is generated as follows. There are 25 nodes in the network, and they are confined in a square area of 1000 m by 1000 m. The transmission range of a node is 250 m. A modified \way-point" movement model is used to model the random movement of the nodes [37]. In the beginning, the nodes are randomly placed in the area. Each node remains stationary for a pause time, the duration of which follows an exponential distribution with a mean of 10 seconds. The node then chooses a random point in the area as its destination and starts to move towards it. The speed of the movement follows an uniform distribution between 0 and the maximal speed v. Network mobility is varied when we change v. Di erent network scenarios for v= 0, 5, 10 m/s are generated. An example of the topology of this network is given in Figure 3.4. The scenario v = 0 represents a static network with no link change. At v = 10 m/s, on average a node experiences a link change every 5 seconds. After reaching a destination, a node pauses again and starts to move towards another destination as previously described. This process is repeated for the duration of the simulation (300 seconds). The only constraint of the movement pattern is that it does not cause network partitions. Without network partition, there is always a route from a source to a destination, so no packet is dropped because the destination is unreachable. All dropped packets are due to network congestion or temporary route failure. When the movement pattern is generated, caution is taken to prevent network partition. If a partition occurs, the node causing the partition randomly picks another destination and starts to move towards it. The node does not pause in this case. An example of this network is a group of soldiers moving on foot in a loose formation. Changes in their relative positions are modeled by this movement pattern. In order for the leader to issue command to his soldiers, no one is allowed to stray away, therefore no partition occurs in the network. User tra c is generated with constant-bit-rate (CBR) sources, where the source and the destination of a CBR session are chosen randomly among the nodes. During its lifetime of 30 seconds, a CBR source generates 20 packets per second. A CBR source does not adjust its transmission depending on the network

congestion, and all 600 packets are always transmitted irrespective of how many of them get through. There is no admission control for a CBR source. The size of a CBR packet is 64 bytes, and it becomes 84 bytes after an IP header is added. A packet is transmitted in three time slots. The starting time of a session is randomly chosen between 0 to 270 seconds, so a session always ends naturally by the end of the simulation. The o ered tra c load is varied by increasing the number of CBR sessions generated during the simulation from 20 to 360. Ten di erent tra c patterns are generated and their simulation results are averaged. We measure the number of packets received by the destinations and the average packet delay. We also measure the number of sessions that are serviced and average packet delay for these serviced sessions. A session is called "serviced" if at least 90% packets are received by the destination ^{k}. This is a measurement of the quality-of-service provided to the end user (the application layer).

3.4.3Simulation results

We first investigate how frequently E-TDMA should update the schedules. The parameter K, the number of information frames between two control epochs, determines how often the schedules get updated. With the number of temporary color $N = 1$, there could be at most one node in a two-hop neighborhood to make new slot reservation in a control epoch. The frequency with which an average node can make reservations is much lower than the frequency the control epoch is executed. It is important for E-TDMA to upgrade the schedules as frequently as possible, provided that it does not incur too much overhead. A smaller K leads to more frequent schedule update but heavier scheduling overhead; a larger K leads to less frequent schedule update but lighter overhead. However, by choosing a large K and generating less overhead, one does not always achieve higher network throughput. This is due to nodal movement. When nodes move, collisions arise in the schedules, and E-TDMA responds slowly with less frequent schedule updates. This leads to more and longer-lasting conflicts in the schedules and reduces packet throughput. When nodes move faster, a smaller K becomes more desirable. Unfortunately E-TDMA does not have a means of changing the parameters dynamically. One can only choose a K that works well under certain conditions. We experiment with di erent K (4, 8, 16) under medium mobility ($v = 5$ m/s) and choose the best, and use this K for the rest of the simulations.

Figures 3.5 and 3.6 show the packet throughput and the average delay for K = 4; 8; 16. The schedule update frequencies (scheduling overheads) are 17 Hz (14%), 9.4 Hz (7.6%), 4.9 Hz (3.9%) respectively. We nd K = 4 achieves higher throughput and lower delay than the others, so K = 4 is used for the rest of the simulations.

Figures 3.7 to 3.8 show the packet throughput and average packet delay of E-TDMA and 802.11 under di erent tra c loads and node speeds. The number of permanent color 12 is chosen based on the maximal nodal degree encountered in the simulation (11). We start by looking at the immobile case (v = 0). When the network is static, once a slot is reserved it remains conflict-free. So this is the ideal case for E-TDMA. When the network tra c is light, both protocols deliver almost all the packets. The packet delay is much lower with 802.11, because under low tra c there is little collision, and a packet is usually transmitted suc-cessfully right away. With E-TDMA a slot has to be reserved rst which causes a non-negligible delay. When tra c gets heavy, more collisions (and backoffs) take place with 802.11, and the throughput reaches its saturations. Beyond a threshold, packet delay increases dramatically. With E-TDMA, every transmis-sion is collision-free, which means its packet throughput increases steadily until every slot is reserved. Average packet delay with E-TDMA only increases slowly with o ered tra c. Because a CBR source always transmits at the same rate, under heavy tra c E-TDMA cannot reserve enough slots. The network becomes over-loaded and packets are delayed and dropped. Compared with 802.11, E-TDMA is more susceptible to nodal movement. When nodes start to move, a slot reserved by E-TDMA can be corrupted and packet collisions take place. An E-TDMA node needs to contend again if it loses an information slot. It is also possible that the permanent color of a node becomes corrupted and has to be discarded. Before this node reserves another permanent color, it experiences a "black-out" and collision could happen in its schedule. Every session going through this node is a ected. When this happens the routing protocol needs to nd another route not using this node. In contrast, the 802.11 protocol does not maintain any channel state and the medium is acquired by RTS/CTS ex-change for every packet. Mobility is handled only at the network layer. When network topology changes and a link breaks, the routing protocol reacts quickly by changing to a di erent route. Such a rout change, and the resulting

changes in bandwidth, are handled easily by 802.11 and AODV. In comparison, these changes are handled poorly by E-TDMA, especially with the original AODV protocol (results of E-TDMA with original AODV can be found in Chapter 5). As a consequence, E-TDMA degrades with node speed v more quickly, both in terms of packet throughput and packet delay. It can be expected that when node speed v increases further E-TDMA will become inferior to 802.11 and break down at some point.

When compared at the session level (Figures 3.9 to 3.11), behavior of the two protocols becomes di erent from that at the packet level. Although the packet throughput of 802.11 saturates when tra c gets heavier, the corresponding ses-sion good-put decreases. This is because with 802.11, every packet is transmitted in the channel on an equal basis, and is equally likely to be dropped when the tra c is heavy . As more packets are dropped from all the sessions, fewer ses-sions have 90% or more packets delivered, thus the session good-put decreases. With E-TDMA, a session which has its bandwidth reserved is guaranteed of its throughput, therefore not a ected by network congestion. The session good-put is kept high under heavy tra c. In the meantime, the session good-put drops faster with node speed than the packet throughout. Compared with $v = 0$, at $v = 5$ m/s only half as many sessions are serviced, and at $v = 10$ m/s only one third of the sessions are serviced. This is because once a session is broken by some topological change, it may not restore its time slots , or the delay of doing so is too long, and more packets are dropped. This is not a problem with 802.11. In fact when nodes move, under light tra c the session good-put is actually lower with E-TDMA than with 802.11, due to the delay to restore the corrupted time slots. A serviced session often su ers little disturbance during its lifetime, and its packet delay is well below the average delay of all packets (Figure 3.10). It is clear that 802.11 is better for light tra c and highly mobile networks; E-TDMA is better for heavy tra c and less mobile networks.

Figures 3.12 and 3.13 show the packet delay and jitter of a session in E- TDMA under light tra c condition. Packet jitter J_i is calculated using the RTP de nition:

$$D_{i-1,i} = (R_i - S_i) - (R_{i-1} - S_{i-1})$$

$$= (R_i - R_{i-1}) - (S_i - S_{i-1}); \quad (3.4)$$

$$J_i = \frac{15}{16} J_{i-1} + \frac{1}{16} D_{i-1,i}; \quad (3.5)$$

where $D_{i-1,i}$ is the di erence between the transmission time of packet i and $i - 1$, S_i and R_i are the time packet i is sent or received, respectively. When the transmission rst begins, packets experience long delay. After a route is found and the bandwidth on the route is reserved, the packets are transmitted in the reserved time slots and experience short delay. In the middle of the transmission, the route breaks and packets are lost. Transmission is restored after a new route is found. The new route is one hop longer than the original one, therefore packets experience longer delay. Note that the delay is not smooth even when the route is not broken and enough time slots are reserved. This is because the arrival of the data packets is not synchronous with the time slots reserved. Di erent packets have to wait for di erent time before their transmissions start. Because both the packet arrival and the reserved time slots are periodic or pseudo-periodic, the packet delay and the packet jitter exhibit some degree of periodicity. This is clear from the insertion of Figure 3.12. When the tra c gets heavy, a session may not be able to reserve all the time slots it needs, or may not be able to restore its time slots after its route breaks. Consequently the packet delay and packet jitter degrade with tra c.

We also simulated the two protocols in a larger network with 40 nodes in an area of 1250 m by 1250 m. The movement patterns of this network are generated in the same way described early. An instance of the topology of this network is shown in Figure 3.4. The average nodal density of this network is the same as that of the smaller network, but maximal nodal degree is higher. The number of permanent colors of E-TDMA is increased from 12 to 15 to accommodate this. Figures 3.14 to 3.17 show the packet throughput, average packet delay, packet dropping probability, session good-put, average packet delay of serviced sessions in this larger network, respectively. The results are similar to those in the smaller networks, except that mobility now

takes a heavier toll on E-TDMA. This is because in the larger network, a packet needs to travel more hops to reach its destination. Although the cost of E-TDMA for reserving time slots on a single hop remains the same, the cost of reserving time slots from end to end on the entire route increases with the route length. The longer the route, the more di cult to reserve and to maintain time slots on the entire route, especially when the nodes move. The session throughput drops by 72% and 84% respectively at $v = 5$ and $v = 10$ m/s relative to $v = 0$. How to provide session QoS by making slot reservations in a large mobile network is still an open problem.

Besides CBR tra c, we also tried with tra c of variable transmission rate using TCP. Unfortunately E-TDMA and TCP do not work well together. Be-cause a TCP agent adjusts its transmission based on its throughput with the sliding window scheme, its transmission rate varies with time. E-TDMA has di culty calculating and reserving a stable bandwidth for a TCP session. This couples with the positive feedback nature of TCP and the resulting throughput is much lower than 802.11. Packet delay is also longer. More work is needed if E-TDMA is to be used to carry TCP tra c.

3.5 Conclusions

A new protocol for generating and maintaining conflict-free TDMA transmis-sion schedules for mobile ad hoc networks has been developed. This protocol is based on the idea of frequently updating the current TDMA schedules on a local basis by many nodes in many parts of the network simultaneously. It is in fact a hybrid scheme which uses contention to determine the set of nodes which can make new slot reservations at an instance. By using contention, the operation of a node is only a ected by those nodes in its two-hop neighborhood and is insensitive to the network size. Therefore the protocol is scalable and can be used for large or dynamic networks. The schedules of the entire network evolve over time to accommodate changes in both the network topology and in the bandwidth requirements. E-TDMA is unique in that it uses a separate, dy-namically maintained broadcast schedule (*ctrl schedule*) to exchange scheduling information between the nodes, and uses limited contention for signaling; in the schedule used for user data transmission (*inf o schedule*), a node can reserve and mix di erent kind of

transmissions (unicast, multicast and broadcast) freely. It is designed for heavy tra c under low to medium network mobility. A limitation of this protocol is that its parameters are fixed and needed to be estimated a priori; a fixed set of parameters work well only within a certain range in terms of nodal density and nodal mobility. It is desirable that these parameters can be dynamically adjusted based on the real network situation. The performance of E-TDMA has been studied with simulations and is compared with that of the IEEE 802.11 protocol. Simulation results showed that E-TDMA works bet-ter under heavy tra c, producing higher throughput and lower delay; but it degrades more rapidly under nodal mobility, than 802.11. Its application is ul-timately limited by nodal mobility and nodal density. Used with a QoS routing protocol developed in Chapter Five, it can provide better QoS for CBR traffic than 802.11.

3.6 Appendix: Pseudo-code of E-TDMA

Parameters of E-TDMA *f*

number of permanent colors *M* ;

number of temporary colors *N* ;

P C = *f*all permanent colors*g*, *T C* = *f*all temporary colors*g*; *ctrl f rame* = *P C [T C*;

number of information frames *K* in an information epoch; number of information slots *L* in an information frame; *inf o f rame* = *f*all information slots*g*;

life time of a neighbor node *nb ttl*;*g*

Data structure E-TDMA *f*

Data maintained at a node *f*

my id;

my ctrl schedule;

my permanent color = *fc 2 P C; state*(*c*) = *T ransg*; *my temporary color* = *fc 2 T C; state*(*c*) = *T ransg*;

/* a node has at most 1 permanent and 1 temporary color */ *my inf o schedule*;

information about a slot (color) *s* in *my inf o*(*ctrl*) *schedule* is referred to as *my state*(*s*) and *my target*(*s*);

a list *N B* of 1-hop neighbors and their schedules, where an entry contains: (*id; ctrl schedule; inf o schedule; exp time*);

information about a neighbor n_i *2 N B* in a slot *s* is referred to as:

N B(n_i) *! state*(*s*);

N B(n_i) *! target*(*s*);

N B(n_i) *! exp time*;

g

Information contained in a schedule-update packet (*su packet*) *f* (*id*, *ctrl schedule*, *inf o schedule*);

g /* A *su packet* should be encoded in a bandwidth-e cient way */

/* A node resets its states and *N B* list when it is rst turned on */ *f unction* node initialization() *f*

N B = ;;

for (*8s 2 ctrl f rame [inf o f rame*)

f my state(*s*) = *Idle*; *g*

g

/* A node contends for a temporary color in a contention phase for */

/* permission to reserve a permanent color or information slots */ *f unction* contention phase() *f*

if (*my permanent color* = *; jj* need new information slots) *f* contend for a temporary color with the FPRP protocol; if (successful to acquire a temporary color *tc 2 T C*) *f*

my state(*tc*) = *T rans*;

my target(*tc*) = *Broadcast*;

g

g

g

/* In the allocation phase a node updates its schedules */ *f unction* allocation phase() *f*

at the beginning of *A* phase *f*

for (*8n*$_i$ *2 N B; N B*(*n*$_i$) *! exp time < current time*) *f* delete *n*$_i$ from *N B*;

for (*8s 2 ctrl f rame [inf o f rame*, (*my state*(*s*) = *Recv-jj my state*(*s*) = *T rans*) && *my target*(*s*) = *n*$_i$) *f* check passive slot(*s*);

g

g

release unused information slot();

g /* delete obsolete neighbors and release unused slots */

in a slot *c f*

if (*c* = *my permanent color*) *f*

broadcast *my ctrl schedule* and *my inf o schedule* in a *su packet*; *g* /* transmit schedules to the neighbors */ else if (*c* = *my temporary color*) *f*

if (it is the c^{th} *A* frame) *f*

if (*my permanent color* = *;*)

f reserve permanent color(); *g*

if (*my permanent color 6*= *;* && need a new information slot)

f reserve infomation slot(); *g*

g /* make new reservations */

broadcast *my ctrl schedule* and *my inf o schedule* in a *su packet*;

g

else *f* /* listen for the schedules of others */

listen for any incoming *su packet*;

if (receive an error free *su packet* from node n_i) *f* if (n_i *62N B*) *f*

add n_i to *N B*;

g /* add a new neighbor */

copy *ctrl schedule* in *su packet* to *N B*(n_i) *! ctrl schedule*; copy *inf o schedule* in *su packet* to *N B*(n_i) *! inf o schedule*; *N B*(n_i) *! exp time* = *current time* + *nb ttl*;

update my schedule();

g /* update the schedules based on this packet */ if (receive a packet with error) *f*

my state(*c*) = *Collision*;

g /* this color is now has a collision */

g

g

at the end of *A* phase *f*

if (*my temporary color 6*= *;*)

R *my state*(*my temporary color*) = *Idle*; *g g* /* invalidate the temporary color */

g

/* Reserve a permanent color in *ctrl schedule* */ *f unction* reserve permanent color() *f*

if (*9* a color *c 2 P C; my state*(*c*) = *Idle*) *f*

my state(*c*) = *T rans*;

my target(*c*) = *Broadcast*;

g /* when there are more than one *c*, choose one randomly */

g

/* Reserve an information slot in *inf o schedule* */ *f unction* reserve infomation slot() *f*

for every new required information slot to transmit to *R* *N B f*

if (*9* a slot *s 2 inf o f rame;* ((*my state*(*s*) = *Idle jj my state*(*s*) = *Block r*)

(*N B*(n_i) *! state*(*s*) = *Idle jj N B*(n_i) *! state*(*s*) = *Block t*,
*8*n_i *2 R*)) *f*

my state(*s*) = *T rans*;

my target(*s*) = *R*;

g /* when there are more than one *s*, choose one randomly */

g

g

/* Update my *ctrl schedule* and *inf o schedule* based on my neighbors */ *f unction* update my schedule() *f*

for (*8s 2 ctrl f rame [inf o f rame*) *f*

if (*my state*(*s*) = *T rans*)

f check transmission slot(*s*); *g*

else

f check passive slot(*s*); *g*

g

g

/* Check a slot when I transmit */

f unction check transmission slot(*s*) *f*

for (*8n_i 2 my target*(*s*)) *f*

state_i = *N B*(*n_i*) *! state*(*s*);

target_i = *N B*(*n_i*) *! target*(*s*);

if (*state*$_i$ = *Collision jj state*$_i$ = *Block r jj*

state$_i$ = *Block tr jj* (*state*$_i$ = *Recv* && *target*$_i$ 6= *my id*)) *f* check passive slot(*s*);

g /* stop transmission when error occurs */

g

g

/* Check a slot when I do not transmit */

f unction check passive slot(*s*) *f*

num trans neighbor = 0;

num trans to me = 0;

num recv neighbor = 0;

for (*8n*$_i$ *2 N B*) *f*

if (*N B*(*n*$_i$) *! state*(*s*) = *T rans*) *f*

num trans neighbor + +;

if (*my id 2 N B*(*n*$_i$) *! target*(*s*) *jj*

N B(*n*$_i$) *! target*(*s*) = *Broadcast*) *f*

my target(s) = *n*$_i$;

num trans to me + +;

g

g

else if (*N B*(n_i) *! state*(*s*) = *Recv* &&

my id 62N B(n_i) *! target*(*s*))

f num recv neighbor + +; *g*

g

if (*num trans to me* > 0) *f*

if (*num trans neighbor* > 1)

. *my state*(*s*) = *Collision*; *g*

else

. *my state*(*s*) = *Recv*; *g* return;

g

if (*num recv neighbor* 1 && *num trans neighbor* 1)

my state(*s*) = *Block tr*; *g*

if (*num recv neighbor* 1 && *num trans neighbor* = 0)

f *my state*(*s*) = *Block t*; *g*

if (*num recv neighbor* = 0 && *num trans neighbor* 1)

f *my state*(*s*) = *Block r*; *g*

if (*num recv neighbor* = 0 && *num trans neighbor* = 0)

f my state(*s*) = *Idle*; *g*

g

/* Stop transmissions in slots I do not need */ *f unction* release unused information slot() *f*

for (*8s 2 inf o schedule; my state*(*s*) = *T rans*) *f*

if (*s* is no longer in use)

f check passive slot(*s*); *g*

g

g

/* Provide information about one-hop neighbors */ *f unction* get 1 hop neighbors() *f*

return *N B*;

g

/* Provide information about two-hop neighbors */ *f unction* get 2 hop neighbors() *f*

N B_2 = ;;

for (*8*n_i *2 N B*) *f*

for (*8c 2 ctrl f rame*) *f* if (*N B*(n_i) *! state*(*c*) = *Recv* &&

N B(n_i) *! target*(*c*) *62N B [my id*) *f*

add *N B*(n_i) *! target*(*c*) to *N* B_2;

g

g

g

return *N* B_2;

g

/* Transmits or receives in an information slot according to *inf o schedule* */ *f unction* information slot(*s*) *f*

if (*my state*(*s*) = *T rans* && *my target*(*s*) = *R*) *f*

transmit an information packet (or a fragment thereof) to *R*;

g

else if (*my state*(*s*) = *Recv* && *my target*(*s*) = n_i) *f*

listen for an incoming information packet *inf o packet* from n_i; if (*inf o packet* is error free) *f*

pass *inf o packet* to upper layer;

g

else if (*inf o packet* has error) *f*

my state(*s*) = *Collision*;

drop *inf o packet*;

Chapter 4

Distributed channel probing and dynamic channel allocation for wireless networks

4.1 Introduction

Power control (PC) and dynamic channel allocation (DCA) are two e ective means to improve the capacity of a wireless network [14, 9, 15, 53, 54]. By com-bining the two together, one can expect the network capacity to increase further. However, an important problem is how to characterize channel utilization and how an algorithm can use such information to facilitate channel selection. Most schemes which combine DCA with power control use interference power as a criterion for channel selection [55, 16, 56]. In these schemes, when a user needs to choose a channel for its transmission, the corresponding receiver measures the interference power in all (or a subset of) the channels, and the channel with the lowest interference power is selected. The logic behind is that the interference power measured at a receiver is proportional to the transmission power of all the other transmitters, and is an indication of the \crowdedness" of the channel. By choosing the least crowded channel, the new user will have a better chance for being admitted, and the required transmission power in this channel is likely to be lower than in other channels. However, a fact neglected here is that the chan-nel condition changes when a user starts transmission in a channel. This new transmission is a source of interference to other on-going transmissions sharing the same channel, and the channel condition changes as the other transmissions increase their own powers to compensate for the additional interference. Better channel selection can be made if this channel dynamics is taken into account. Recently, channel probing has been proposed for wireless networks [17, 57, 18]. A channel probing schemes require a new user to monitor the response of other co-channel users, often in terms of the interference power, as it is increasing its transmission power, and to estimate the channel condition accordingly. With channel probing, it is possible to perform predictive/interactive admission con-trol. This provides a way to better protect active users as well as to make better

channel selection for new users. Channel probing is usually more complicated than traditional schemes and requires more overhead, but it has the potential to achieve higher network capacity and deserves further investigations

Channel probing was rst introduced in [17], as part of the DCA-ALP con-trolled power update algorithm for protection of active users. In DCA-ALP, a new user increases its transmission power gradually. It can estimate the channel admissibility from its signal to interference ratio (SIR) measurements in the rst few power-up steps. A user can also predict the required transmission power and the number of iterations required to reach its target SIR. Active users are pro-tected from the new user at all time. The scheme in [18] is designed to provide a fast probing mechanism. The channel probing is completed in one step instead of multiple steps. In the \Soft and Safe Admission Control" scheme [57], although a user does not predict its admissibility, it gradually increases its transmission power until it is either admitted or rejected. With the exception of [57], these channel probing schemes are fully distributed and require no global coordination. Users only interact with each other by causing and measuring the interference in the channel. Because di erent users do not coordinate their probings, it is possible for multiple users to simultaneously probe a channel without knowing the activities of the others. This problem has not been addressed before and is studied the rst time.

We introduce a new channel probing scheme which allows a user (transmit-ter), in co-operation with the corresponding receiver, to probe a channel, to estimate the channel condition and to further predict the required transmission power to meet its desired SIR. It is a fully distributed scheme which requires no communication between di erent transmitter/receiver pairs, yet it is capable of handling the case where a channel is being probed by multiple users simultane-ously. The local admissibility of each users, estimated from probing the channel, is equivalent to the global feasibility calculated with information of the entire network. By probing the channels, a user can choose the best channel. Hence the channel probing scheme can be used to improve the performance of dynamic channel allocation scheme. The predicted transmission power can be used as the criterion for channel selection. This scheme is compared with other channel al-location schemes via simulation. The simulation results show that with the new scheme, newly-

arrived transmissions experience less blocking and the on-going transmissions su er less disruptions.

4.2 The system model

We consider a TDMA (or FDMA)-based wireless network where the transmitters can adjust their transmission power continuously within a given range. Each time slot in a TDMA time frame (or a carrier frequency in a FDMA system) is referred to as a channel. Inter-channel interference is not considered here, but can be included if necessary. Nodes perform a closed loop power control algorithm described as follows. The power control algorithm used is the same as that in [14, 9]. Suppose that there are M active links, labeled 1 through M, in a given channel. Each link i consists of a transmitter and a receiver, and has a target SIR γ_i^t. Di erent links may have di erent target SIRs. We assume this transmitter/receiver pair is determined by some other schemes, and it is considered fixed here. The terms *link* and *transmitter=receiver pair* are used interchangeably, and transmission power and SIR of a link respectively means the output power of the transmitter and the SIR at the receiver. Let $g_{i;j}$ be the propagation gain between the j*th* transmitter and the i*th* receiver, and $G^M = [g_{i;j}]_{M;M}$ be the transmission gain matrix of the system. To keep it simple, we assume that $g_{i;j}$ is a positive constant and only depends on the location of the two nodes, although in fact it su ers various kinds of fading and is stochastic in nature. Therefore we assume that all the users are static and the propagation gain $g_{i;j}$ is time-averaged. The SIR of a link i is determined by the transmission powers of the active links, the transmission gain, the target SIR and the noise n_i at the receiver.

This power control algorithm converges at a geometric rate, which is determined by $\rho(D^M Z^M)$ [15]. Except for the case where $\rho(D^M Z^M)$ is close to 1, the

convergence is fast, and the error becomes small enough after a small number of iterations. The M links are called admissible if they can all achieve their target SIRs, and inadmissible otherwise. In the latter case the system is called \interference-limited", because the interference cannot be overcome simply by increasing the transmission power. When the maximal transmission power is taken into consideration, it is also necessary that

$$P^M \quad p_{max}1^M ; \tag{4.6}$$

where 1^M is the all 1 (column) vector with length M . If condition 4.4 is satis ed, but the transmitters do not have enough power, the system is called \power limited". Such a system can be made admissible by increasing the maximal transmission power.

4.3 The channel probing algorithm

The proposed channel probing mechanism is based on the assumption that the set of active links update their transmission power frequently, and will react to increased interference in the channel quickly by increasing their own power levels. When a set of new links join the channel and start to transmit, these active links experience additional interference, and as a consequence, will raise their powers accordingly. Their power increase is proportional to the power of the new links. If the new links transmit their signals at a prede ned power level and measure the corresponding SIRs, they can estimate the channel condition. This is termed \channel probing". These new links, by probing a channel, can predict whether the channel is admissible and, if the answer is yes, what is the required transmission power. To simplify the analysis, we ignore the maximal

power constraint in the next two sections, and assume the transmitters always have enough power. The effect of limited p_{max} will be discussed later. The details of the channel probing algorithm is given as follows.

Suppose a set of M links, 1 to M, are already transmitting in a channel, and they apply power control and have achieved their SIR balance with target SIR t_M . Their transmission power vector is given by links, $v^M = [v_1; v_2; \ldots; v_M]^0$ is their (thermal) noise vector, E^M is an extraneous noise vector introduced by any other interferences, and $P^M = [p_1; p_2; \ldots; p_M]^0$ is their transmission power vector. Initially $E^M = [0; 0; \ldots; 0]^0$. When a set of new links ($M + 1$ to $M + N$) start to

transmit in the same channel with transmission power vector $P^N = [p_{M+1}; \ldots; p_{M+N}]^0$, they cause additional interference to the M existing links

Note that the power increase is proportional to the transmission power P^N of the new links. The SIR of a new link k, $M+1 \; k \; M+N$, is given by is the (normalized) noise and interference power at receiver k before the new links emit any power, and $k;j$ is given by

$[z_{j;1}; z_{j;2}; \ldots; z_{j;M}]$: Note that each component of B^N is positive, and B^N is an all positive matrix. The positivity of B^N will play a major role later. Matrix B^N represents the interference among the N new links. It consists of two parts: the direct interference through propagation gain matrix (Z^N) and the indirect interference through the M active links. If these N new links update their transmission powers and achieve their target SIRs, their transmission powers are given by

The N new links and the M existing links can achieve their target SIRs if and only if the condition $(I - {}^t_N B^N)^{-1} > 0$ is true, or equivalently, $P({}^t_N B^N) < 1$. This is shown as follows

Proposition 4.1: The channel is feasible for all the M active links as well as the N new links if and only if $P({}^t_N B^N) < 1$, where $P({}^t_N B^N)$ is the Perron eigenvalue of the matrix ${}^t_N B^N$.

Proof: The channel is feasible for the $M+N$ links i $(I - {}^t_{M+N} Z^{M+N})^{-1} > 0$, where Z^{M+N} is the propagation matrix associated with the $M+N$ links, and ${}^t_{M+N} = diag(\gamma_1^t; \ldots; \gamma_M^t{}_{+N})$. Rewrite Z^{M+N} as Feasibility for the $M+N$ links requires each of the $C_{11}; C_{12}; C_{21}; C_{22} > 0$. The fact that the M active links transmit in the same channel implies $(I - {}^t_M Z^M)^{-1} > 0$. Inspecting the four C terms shows the inequality $(I - {}^t_{M+N} Z^{M+N})^{-1} > 0$ holds i

Hence studying the feasibility condition for the matrix ${}^t_N B^N$ is equiva-lent to studying the feasibility condition for the matrix ${}^t_{M+N} Z^{M+N}$. If either matrix is known, we can check the feasibility condition $P({}^t_N B^N) < 1$ or $P({}^t_{M+N} Z^{M+N}) < 1$ and calculate the required transmission power, and the entire problem is solved. However, it is very di cult, if not impossible, for the individual links to calculate (or estimate) the entire Z^{M+N} or B^N matrix in a distributed fashion. The following channel probing scheme is proposed as a way for the individual links to estimate the feasibility of the channel:

Each new link probes the channel by transmitting a probing signal, or \prob-ing tone", with transmission power $_{p}^{ps}$, and measure the corresponding SIR. The probing signal can simply be a prede ned training sequence, or can carry some basic information. All the probing nodes transmit with the same $_{p}^{ps}$, and $^{P}{}^{N} = {}_{p}^{ps}1^{N}$, where 1^{N} is the all 1 column vector of length N. The SIR of linkduring probing, $_{k}^{p}$, is given by its receiver *before* and *during* transmission of the probing signal. Let $_{p}^{r}{}_{k}$ (0) be the received power (noise and interference) at receiver k before the new links probe the channel, and $_{p}^{r}{}_{k}$ ($_{p}^{ps}$1) be the received power (signal plus noise and interference) when the links are probing the channel. By de nition, k is the normalized version of $_{p}^{r}{}_{k}$ (0), where the normalization factor $_{k;k}$ can be obtained as

4.6 Probing based channel allocation

For a given set of links and a number of channels in a TDMA/FDMA system, nding a good channel assignment is a di cult problem. The channel probing scheme provides a simple, yet effective means to do so.

When a node needs to nd a channel and transmit to another node, the two nodes can pair up as a link and perform channel probing. This new link can probe all (or some) of the channels, and determine which channels are available and also predict their transmission powers. It can choose the channel requiring the lowest power, thus saving energy as well as reducing the interference in the channel. This way more links can be admitted into the system, thereby increasing the network capacity, or the transmission power of the links can be reduced, thereby enhancing the battery life. Although the channel probing scheme cannot always predict the transmission power accurately, the case of a single arrival (N = 1) is most common in a system of modest size and low arrival rate, and the predicted transmission power is accurate.

We study the following channel allocation schemes and compare their per-formances. The rst scheme is \random channel selection" (RCS). When a link looks for a channel, it chooses a

channel randomly and starts to power up. The second scheme is called \sensing-based channel selection" (SCS). It di ers from RCS in that when a link looks for a channel, the receiver measures the inter-ference and noise power in all the channels, and chooses the channel with the lowest interference level. It is similar to the scheme for channel selection used in [55, 16, 56] and is commonly found in the DCA literature. With the notation in Section 3, the channel with the lowest is selected. In \probing-based chan-nel selection" (PCS), a link probes all the channels, and picks the one with the lowest predicted transmission power. If all the channels are inadmissible, the link is blocked without trying to power up in any of the channels. This way the interference caused to other links is reduced signi cantly. On the other hand, in RCS and SCS, a link learns its inadmissibility to a channel "the hard way", and can cause excessive interference to on-going transmissions and force some transmissions to be dropped. The di erence between the three channel allocation schemes stops here. Once admitted into a channel, a link applies power control and tries to maintain its target SIR, until its transmission ends and it releases the channel, or its SIR is consistently lower than the target and it deems the current channel unavailable. In the second case, if the link is new to the channel, it stops its transmission and is blocked from the system. If the link is an old link in the channel (has been active for sometime), before it is dropped from the system, it tries to nd another feasible channel, using the same scheme as a newly arrived link. It is lost when it fails to nd an admissible channel after a number of trials. For the special case when the number of channels is one, the channel al-location schemes degenerate into rules of admission control. The PCS scheme becomes \probing-based admission control" (PAC), where admissibility is deter-mined rst by channel probing. The RCS and SCS become the same scheme, since there is only one channel. Without probing, a new link is only blocked from the channel after it tries to power up and fails, and there is indeed no admission control. For this reason RCS and SCS become \null admission control" (NAC) in a single-channel system

The performance of the channel allocation schemes are measured in terms of the blocking probability of newly-arrived transmission requests (P_b), the forced dropping (termination)

probability of on-going transmissions (P_d), the probabil-ity that an on-going transmission is forcefully relocated to another channel (P_r), and the average transmission power of the links. Although forced termination of an on-going transmission is the most unfavorable case, a transmission forcefully relocated to a di erent channel su ers temporary link quality degradation, and P_r reflects the disturbance experienced by on-going transmissions. The channel allocation schemes are evaluated via simulation in the next section.

4.7 Simulation studies

Transmission power p_{max} = 1 W. A power update interval (PUI) is de ned as the time required for a link to measure its SIR and update its transmission power accordingly. Following [62], the length of a PUI is taken to be 200 ms. All active links update their transmission powers every PUI. For simplicity, we assume all the active links update their transmission powers synchronously, although the asynchronous version of the power control algorithm converges as well [58]. We assume a receiver transmits its SIR measurement to its transmitter through a separate channel, and no delay or error is incurred. Network tra c, arriving at the individual links (single hop), consists of voice calls and has exponential inter-arrival and service times. The service time has a mean of 120 seconds. The o ered load is controlled by varying the expected inter-arrival time of new call requests to each link. There is no new arrival to a link which is already admitted and undergoing transmission. The target SIR is γ^t = 16 dB for all the links. If an active link nds its SIR below the target for 2 consecutive seconds, it is forced to withdraw from its current channel and starts to look for a new one, using the same scheme as a newly-arrived link. It is dropped from the system when it fails to nd a valid channel after 2 trials. A new link is blocked immediately when it fails to reach its SIR in the channel it selects. It is not given a second chance. There is no communication between di erent links (hence no distress signal). Di erent links only interact through the interference they cause to each other.

In the channel probing scheme, the probing power p^{ps} = 0.1 mW. The average SIR of the probing signal is approximately 4dB. When a new link probes the channel, it uses $\gamma^n = \gamma^t$. No

SIR penalty for the new links is applied. When a channel is being probed by some new links, the transmission power in the channel increases by about 15%. A probing signal must last long enough to allow other links to react fully. In the simulation a probing signal has a duration of 5 PUI (1 second). Because it is shorter than the time for an active link to withdraw from a channel due to link quality degradation (2 seconds), it is not likely that an active link is forced out of its channel by probing signals.

In the experiments, 100,000 calls are simulated for each case and the re-sults are shown in Figures 4.1 to 4.4. As expected, the RCS algorithm works worst in all the performance measures. Because no attempt is made to select a good channel, a newly arrived call has a high blocking probability. Choosing the wrong channel not only causes new call requests to be blocked, but also causes signi cant disturbance to on-going transmissions and causes high reloca-tion probability and high dropping probability. Under heavy load the average transmission power is actually lower in RCS than SCS and PCS, because higher blocking and dropping probability results in fewer transmissions. Between the other two schemes, overall the PCS algorithm outperforms the SCS algorithm. It has a lower blocking probability as well as a lower relocation probability. The ability to take into consideration the response of the active link ( ), in addition to the current interference ( ), provides a better way for channel selection. The term is not only more important than, it becomes dominant as the interfer-ence increases and solely determines the feasibility of a channel. The ability of PCS to detect the inadmissibility of a link before it fully powers up and causes excessive damage to other links is extremely valuable. In the SCS algorithm, active links often have to switch to other channels, when new links force their way into the system. Link relocation is much less frequent in the PCS algorithm, which means on-going transmissions are less disturbed. However, frequent link relocation provides a means of load-balancing and has its bene t. As the links are re-shufied more frequently in the SCS algorithm, tra c hot spots are elim-inated. This leads to a lower average transmission power in the SCS algorithm than in the PCS algorithm. The dropping probability for active links are roughly the same for these two algorithms. The relative performance of the two algo-rithms does not change much as the tra c varies, so it can be advantageous to use PCS even in a lightly-loaded system. We also

simulated the channel probing scheme in a CDMA network with ran-dom spreading sequence and conventional matched lter receiver. Because the spreading sequence is random, a transmitter can only control its transmission power (or whether to transmit at all). This can be viewed as a single channel system, where the PCS scheme becomes PAC (probing-based admission control), and the RCS and SCS schemes become NAC (null admission control). The gain matrix of this network is the same as the TDMA network. The processing gain is 128 (PG = 128) and target SIR is 9 dB. Due to the wider bandwidth, the receiver noise increases to 10^{-13} W. The maximal transmission power is 1.5 W. Figure 4.5 compares the blocking probability of newly-arrived calls and dropping probability of on-going calls for the PAC and the NAC schemes. Because of the admission control imposed by channel probing, newly-arrived calls experience a higher blocking probability with the PAC scheme than with the NAC scheme. However, with the NAC scheme, no protection is provided for on-going transmis-sions, therefore on-going transmissions su er a high dropping probability. On the other hand, no on-going transmission was dropped with the PAC scheme in the simulation. This demonstrates better protection for on-going transmissions by the channel probing scheme. Therefore the channel probing scheme is also

4.8 Discussion

We now compare our channel probing scheme with other similar schemes. In DCA-ALP [17], the transmission power of a new link is updated in a controlled manner. This controlled power increase, together with a safety margin above the target SIR, protects active links from the new link at all time. A new link can estimate the channel admissibility and required transmission power from its SIR measurements in the rst few power-up steps. The limit of this scheme is that it takes many power update iterations to admit a new user, and the safety margin above the target SIR decreases the network capacity. The channel probing scheme of [18] is made faster by separating the probing segment from the data segment and the probing signal is transmitted in the probing segment only, thus eliminating the need for gradual power increase. A new user probes a channel by transmitting with xed power in the probing

segment and measuring the changes in the interference level. Although similar to the probing scheme proposed here, this scheme is based on heuristics and does not relate the individual link admissibility condition to the global feasibility condition of the channel, and a new link may be falsely admitted or rejected. In these two schemes, channel probing was designed primarily for the nonconstrainted power control case, which could su er under limited transmission power, although [17] uses a distress signal to relieve the problem. The scheme in [57] is designed with the power constraint in mind, and the transmission power constraint dictates how much a new user can increase its power during each iteration. This protects the active users from power outage at all time, and it has the desirable \soft

The channel probing scheme proposed here is closer to [17] than the others. It is too designed without considering the maximal transmission power limit explicitly. The distress signal of [17] can also be applied here to some degree of usefulness. With the current scheme, the time required for channel probing is the same as the time required for the power control algorithm to converge once, therefore it is relatively simple and fast. Here we are concerned with the *long-term feasibility* of all the links in a channel rather than with temporary SIR fluctuation of an individual link. An active transmission will suffer tem- Channel probing used as an admission control scheme in a CDMA network. P_b(NAC) and P_d(NAC) are the blocking probability for new calls and the dropping probability for admitted calls with the NAC scheme respectively. The blocking probability with PAC is P_b(PAC). No on-going transmissions are found dropped with the PAC scheme in the simulations. Temoporary SIR degradation when a new link probes and joins the channel. If it is necessary to maintain the SIR of an active link at all time, the frame structure of [18], which separates the probing segment from the data segment, can be used to make the data transmission immune to the interference from probing. Our scheme allows network heterogeneity, i.e. di erent links may have di erent target SIRs. A link can also adjust its target SIR dynamically based on the current channel condition. While none of the previous

schemes consider the case where multiple links probe a channel simultaneously, our scheme is designed for this multiple probe scenario. We showed the equivalence between the local admissibility learned from channel probing and the global feasibility learned from knowing the entire network (the G matrix) when there are multiple probes. The situation when there is only one new link probing the channel is a special case ($N = 1$) which makes the channel probing more accurate.

In its current form, the channel probing scheme is more applicable in a voice network than in a data network, because the network tra c changes more slowly with voice streams than with bursty data transmissions. If the tra c fluctuates too much in a short time scale, it may be impossible to accurately measure the SIR and apply power control, and the channel probing scheme breaks down. In a system where bursty data transmission takes place, its more appropriate to take a probabilistic approach for power control [63]. The channel probing scheme can be used in an outdoor environment as well as indoors, as long as the propagation gain remains relatively static to allow the power control to converge. For this reason it might be di cult to use the scheme in a mobile environment where the users are moving quickly.

Because the channel probing scheme, as well as the dynamic channel allo-cation scheme utilizing channel probing, is fully distributed and requires little communication between di erent nodes (except that between a transmitter and its corresponding receiver), it is attractive to wireless networks lacking xed infrastructure, such as ad hoc networks, where it is di cult for nodes to com-municate and coordinate with each other. Of course, it can also be applied to a more traditional cellular network. An interesting feature of the current scheme is that not all nodes are required to support channel probing before it can be used in a system. All that is required is that every node can execute the power control algorithm described earlier. For a node which cannot perform channel probing itself, it simply experiences SIR degradation when the channel is probed by another link. When this link adjusts its power to re-achieve its target SIR, it cooperates in the channel probing process of other links without realizing it.

This makes it possible to gradually introduce the channel probing scheme as an add-on feature to some systems which are already deployed.

Currently the channel probing scheme, as well as the power control algorithm, is limited by the time required to measure the SIR (in the order of a fraction of a second). It will become more adaptive if this time can be reduced. In the simulations it is assumed that there is a separate channel to transfer the SIR information from the receiver to the transmitter. This is necessary because the simulated tra c is one-way. In a real network most of the tra c will be two-way tra c, and the SIR information can be piggy-backed to the user tra c, or as part of a control message exchanged between the nodes. Compared with the time scale for SIR measurement, the transmission delay for these messages is very short, and such an approach can be justified.

4.9 Conclusion

A distributed channel probing scheme for wireless networks applying power con-trol has been developed. It allows a new link to estimate the channel condition by transmitting a low powered probing signal. Some important properties have been proven, most noticeable the equivalence between the local admissibility and the global feasibility. The e ect of maximal transmission power has also been discussed. The channel probing scheme can be used as a means of dis-tributed channel allocation for a TDMA or FDMA system, or admission control for a CDMA system. Simulations have shown that it outperforms some other schemes not using channel probing.

Chapter 5

Quality-of-Service routing in mobile adhoc networks

5.1 Introduction

In this chapter the problem of Quality-of-Service (QoS) routing in mobile ad hoc networks is studied. A lot of work has been in the routing area for ad hoc networks. In particular, the Ad-hoc On-Demand Distance Vector routing protocol(AODV) [11], the Dynamic Source Routing protocol(DSR) [12], and the Temporary-Ordered Routing Algorithm(TORA) [13] all demonstrated the ability to route data packets e ciently. They are designed primarily to carry best-e ort tra c in a mobile ad hoc network whose topology changes frequently. At the center of their design is the connectivity between the nodes, or the topology of the network. The routes are calculated solely based on the network topology. Little attention is paid to the amount of bandwidth on the routes, or the end-to-end quality of service delivered to the upper layers. A review of routing protocols for mobile ad hoc networks can be found in [64]. Only recently have people turned their attention to establishing QoS routes in ad hoc networks [65, 66, 67, 68, 69, 70, 71, 72, 73, 74]. QoS routing requires not only to nd a route from a source to a destination, but the route must satisfy the end-to-end QoS requirement, often given in terms of bandwidth or delay. Quality of service is more di cult to guarantee in ad hoc networks than in other type of networks, because the bandwidth resource is usually shared among adjacent nodes due to the wireless medium, and the network topology changes as the nodes move. This requires extensive collaboration between the nodes, both to establish the route and to secure the resources necessary to provide the QoS. The ability to provide QoS is heavily dependent on how well the resources are managed at the MAC layer. Among the QoS routing protocols proposed so far, some use generic QoS measures and are not tuned to a particular MAC layer [69, 70, 73]. Some are designed for a MAC layer which uses CDMA to eliminate the interference between di erent transmissions [65, 66, 71, 74]. Di erent MAC layer models have di erent constraints for successful transmissions, and a QoS routing protocol developed for one type of MAC layer does not generalize to others easily. So far no

work has been done on QoS routing in a flat-architectured, TDMA-based ad hoc network. This is the type of networks studied in Chapter 2 and Chapter 3TDMA transmission is more demanding than CDMA, because transmissions are more likely to interfere. Hence more coordinations among the nodes are required.

We develop a QoS routing protocol for ad hoc networks using TDMA. The object of this protocol is to establish bandwidth guaranteed QoS routes in small networks whose topologies change at low to medium rate. If the nodes move too fast, the QoS approach, which is based on setting up states and reserving bandwidth (time slots) on a route, cannot accommodate the topology changequickly enough and becomes inappropriate. The protocol is based on the AODV protocol, and builds QoS routes only as needed. As a prerequisite to performing QoS routing, one must be able to evaluate the degree of QoS, i.e. the amount of available (residual) bandwidth on a route, thus be able to choose one route which satis es the bandwidth requirement out of many potential routes. We as-sume the application is session-oriented and requires constant bandwidth. With TDMA, time is slotted and bandwidth is allocated in unit of time slots. A ses-sion speci es its QoS requirement as the number of slots it needs on its route. To start with, we rst study how to calculate the available bandwidth on a given route. We will show that to calculate the maximum end-to-end bandwidth is NP-complete, and develop a distributed algorithm for calculating the (non-optimal) end-to-end bandwidth. We will then study how the bandwidth calculation algo-rithm can be used in conjunction with AODV to perform QoS routing. If nodes do not move, once a QoS route is established and the bandwidth are reserved, packets transmitted along this route are guaranteed of bandwidth and through-put. When nodes move, a QoS route is subject to breakage during its lifetime. The proposed QoS routing protocol can also restore a broken route, thus handle node mobility to some degree. Simulations are used to study the performance of this QoS routing protocol.

5.2 The network model

An ad hoc network is modeled as a graph $G = (N; L)$, where N is a nite set of nodes and L is a set of undirected links. A node n_i has a set of neighbors $N\ B_i = fn_j\ 2\ N : (n_i; n_j)\ 2\ Lg$. The bandwidth is partitioned into a set of time slots $S = fs_1; s_2; :::; s_M\ g$ which consists a frame. The transmission schedule of

node n_i is de ned as the set of slots $T\ S_i$ in which it transmits, and the set of nodes R_i^k which is its transmission target set (receivers) in slot s_k , $R_i^k\ 2\ N\ B_i$, $s_k\ 2\ T\ S_i$. With an abuse of notation we will use $T\ S_i$ to refer to both the transmission slots set and the transmission targets sets in these slots. The set $RS_i = fs_k\ 2\ S : n_i\ 2\ R_j^k\ ;\ n_j\ 2\ N\ B_i g$ is the set of slots where node n_i is required to receive from its neighbors. Let $T\ N^{\ k} = fn_i\ 2\ N : s_k\ 2\ T\ S_i g$ be the set of nodes transmitting in slot s_k . A transmission from node n_i to node n_j is labeled as $(n_i\ !\ n_j\ )$, or $(n_i\ !\ n_j\ )^k$ when we want to emphasize it takes place in slot s_k . The schedule of the entire network $T\ S$ is the collection $fT\ S_i : n_i\ 2\ N\ g$. The transmission slots can be assigned by some TDMA slot assignment protocol running at the MAC layer. The details of the slot assignment protocol is not important at the moment, but we assume the following conflict-free property always holds:

If a node n_i transmits in slot s_k $(n_i\ 2\ T\ N^{\ k}\ )$, for every node $n_j\ 2\ R_i^k$, $N\ B_j \setminus T\ N^{\ k} = fn_i g$ and $n_j\ 62 T\ N^{\ k}$.

In other words, if node n_i transmits to node n_j in slot s_k , node n_j itself does not transmit and node n_i is the only transmitting neighbor of n_j in that slot.

Respectively, these are the set of slots when node n_i can transmit without causing interference to its current receiving neighbors (SRT_i), and the set of slots when node n_i can receive without su ering interference from its current transmitting neighbors (SRR_i), given the current transmission schedule $T\ S$. The sets SRT_i and SRR_i are not necessarily the same. This is illustrated in the Figure 5.1. Figure 5.1: De nition of SRR and SRT for TDMA transmissions in an ad hoc network. Suppose there are 2 slots, $S = fs_1; s_2 g$. If the current transmission schedule is $(n_1\ !\ n_2)^1$, $SRR\ 6= SRT$ for nodes n_3 and n_4.

The scenario considered here is session-oriented tra c, where each unidirec-tional session is also called a flow. A request to setup a QoS route for a session is given in terms of < *Source*

Addr; Dest Addr; F low ID; Bandwidth >. We assume a session requires constant bandwidth and tells the routing protocol how many slots it needs. When a QoS route is established for a flow, new slots need to be reserved on the route. These reservations must be conflict-free. From the prospective of nding a QoS route, the sets $fSRT_ig$ and $fSRR_ig$ represent all the constraints presented by the current transmission schedule $T\ S$, because they dictate what slots are in use and what slots are available. For this reason we also express the transmission schedule as $T\ S = fSRT_i; SRR_i; n_i\ 2\ N\ g$.

Given the requirement to establish a session, the QoS routing protocol needs to nd a route with su cient bandwidth, and to determine the set of transmission slots used by each link on the route. This is not easy, because even to nd out the maximum available bandwidth along a *given* route is NP-complete. Without causing confusion the terms *path* and *route* are used interchangeably. We start from the calculation of the end-to-end bandwidth for a given route.

5.3 The bandwidth calculation problem

In order to provide a bandwidth of R slots on a path P , it is necessary that every node along the path nd at least R slots to transmit to its downstream neighbor, and these slots do not interfere with other transmissions. Because of these constraints, the end-to-end bandwidth on the path is not simply the bandwidth on the bottleneck link. The path bandwidth calculation problem, termed $BW\ C$, can be formulated as .

5.4 A bandwidth calculation algorithm

Because the maximum bandwidth for a given path is intractable, we seek alter-natives approximating the optimal solution. Instead of searching for the global maximum, the algorithm developed here only searches for local maximum which ends up to sub-optimality. The attraction of this algorithm is that its sim-ple, iterative calculation, and is well matched to the route discovery mechanism of AODV. It is both computational e cient and produces good results. Two versions of the algorithms will be presented. The forward algorithm ($F\ A$) iter-ates over the hops from the source to the destination, and the backward algo-rithm (BA) iterates from the destination to the source. The terms *f orward* and *backward* only refer to

the direction with which the iteration is carried out. For a given path $P = fn_m ! n_{m-1} ! ::: ! n_0 g$, both calculate the bandwidth from the source n_m to the destination n_0.

5.5 QoS routing

QoS routing requires nding a route from a source to a destination with a certain amount of bandwidth. The bandwidth calculation scheme presented above only provides a method to calculate the available bandwidth for a *given* route. It is *not* a routing protocol, and needs to be used together with a routing protocol to perform QoS routing. The routing protocol chosen here is AODV [11]. AODV is a pure on-demand routing protocol and uses a broadcast route discovery mecha-nism. It relies on dynamically establishing routing table entries. The reason for selecting AODV is that its route discovery mechanism matches the bandwidth calculation scheme very well and is suitable for bandwidth constrained routing. Like AODV, the QoS routing protocol also works on an on-demand basis. A node does not keep routing or bandwidth information it does not need. We start from a short description of AODV. More details of AODV can be found in [11].

5.5.1 The AODV protocol

The Ad-Hoc On Demand Distance Vector (AODV) routing algorithm is a routing protocol designed for ad-hoc mobile networks. It is an on demand algorithm, meaning that it builds routes between nodes only as desired by source nodes. It maintains these routes as long as they are needed by the sources. AODV uses sequence numbers to ensure the freshness of routes. It is loop-free, self-starting, and can also build multicast routes.

AODV builds routes using a route request/route reply query cycle. When a source node needs a route to a destination for which it does not already have a route, it broadcasts a route request (RREQ) packet across the network. The message format of the RREQ is as follows:

6. F lags; Hop Count; Broadcast ID; Source Addr; Source Seq#; Dest addr;

Dest Seq# > : _ _

In addition to the source node's IP address, current sequence number, and broadcast ID, the RREQ also contains the most recent sequence number for the destination of which the source node is aware. The TTL eld in IP packet header is used to control the range to which the RREQ is propagated. The protocol uses an expanding ring search scheme which increments TTL gradually. As a RREQ is forwarded hop by hop and reaches a node, it leaves behind a path from the source. Nodes receiving this packet update their information for the source node and set up backwards pointers to the source node in the route tables. A node receiving the RREQ may send a route reply (RREP) if it is either the destination or if it has a route to the destination with corresponding sequence number greater than or equal to that contained in the RREQ. If this is the case, it unicasts a RREP back to the source. The message format the RREP is as follows:

< F lags; Hop Count; Source Addr; Dest addr; Dest Seq#; Lif etime > :

If a node cannot reply to the RREQ, it rebroadcasts the RREQ. Nodes keep track of the RREQ's source IP address and broadcast ID. If they receive a RREQ which they have already processed (by inspecting the *Broadcast ID*), they discard the RREQ and do not forward it.

As the RREP propagates back to the source, nodes set up forward pointers to the destination. Once the source node receives the RREP, it may begin to forward data packets to the destination. If the source later receives a RREP containing a greater sequence number or contains the same sequence number with a smaller hop count, it may update its routing information for that destination and begin using the better route.

With AODV, at any time there is at most one active route to a given destina-tion in the routing table of a node. This route is used by this node to transmit or forward all the packets addressed to the destination. A route is considered active as long as there are data packets periodically traveling from the source to the destination along that route. Once the source stops sending data packets, the links will time out and eventually be deleted from the intermediate node routing tables. If a link break occurs while the route is active, the node upstream of the broken link propagates a route error (RERR) message to the source node to

inform it of the now unreachable destination(s). A RERR is equivalent to an unsolicited RREP packet with *Hop Count = 1*. After receiving the RERR, if the source node still needs the route, it can re-initiate route discovery.

5.5.2 QoS routing with AODV

Currently AODV provides some minimal control to enable nodes to specify Qual-ity of Service parameters, namely maximal delay or minimal bandwidth, that a route to a destination must satisfy [73]. These QoS parameters, however, are generic and their calculations depend on speci c networks. The QoS measure used here is bandwidth. In a TDMA network, the bandwidth can be calculated using the schemes developed early. A flow is identified by

< Source Addr; Dest Addr; F low ID; Bandwidth > :

To build a QoS route for a flow, the flow information is carried in every AODV routing packet. There could be more than one flows between a source and a destination but with di erent *F low ID*s and possibly di erent bandwidth requirements. These flows do not necessarily share the same route.

Two approaches can be used to build QoS routes. In the rst approach, bandwidth calculation is decoupled from route discovery. With this approach, a route is found by the original AODV protocol without considering the bandwidth constraint. It is bandwidth is calculated afterwards to check whether it has su cient bandwidth to satisfy the QoS requirement. This can be called the decoupled approach, because the interaction between routing and bandwidth calculation is minimal. Decoupled approach is used in INSIGNIA [69], where bandwidth calculation is used for admission control only. In the second ap-proach, called the integrated approach, the bandwidth calculation is performed in conjunction with route discovery, and the routing protocol tries to nd a route with su cient bandwidth. The decoupled approach is presented rst.

5.5.3 The decoupled approach

In the original AODV protocol, bandwidth is not considered when a route is being looked for. A route found may or may not have enough bandwidth to meet the requirement. In the decoupled approach, after a route is found, the amount of available bandwidth on this route is calculated. Only a route with su cient bandwidth can be used. Therefore bandwidth calculation is used for admission-control purpose. In order to nd the end-to-end bandwidth, the calculation needs to be initiated by either the source node in the forward direction with *F A*, or by the destination node in the backward direction with *BA*. We will use *BA* as an example. The algorithm requires the calculation to be done through the entire route. This precludes any node other than the destination to reply to the RREQ and is di erent from the original AODV. This causes increased overhead of route discovery.

The *F A* can also be used. If the source node has a route *P* to the destination and needs to check its available bandwidth, it sends a RREQ which is only forwarded on *P* . The bandwidth calculation is done hop by hop with *F A* and is complete when the RREQ reaches the destination. Since this procedure is very similar to the integrated approach described in the next section, we omit its details here.

Because of the minimal coupling between routing and bandwidth calculation, this approach works with other routing protocols as well. A routing protocol which provides multiple routes will be a better choice than AODV, because the bandwidths along multiple routes can be calculated, thus enhances the chance for nding a route which satis es the bandwidth requirement. However, the decoupled approach is only an admission control scheme and is not a real QoS routing protocol. Here we outline this approach mainly as an example using the *BA* algorithm. We will not proceed further in this direction. A better approach will be to combine bandwidth calculation with route discovery. The result is a fully integrated QoS routing protocol, which is described in the next section.

5.5.4 The integrated approach: a QoS routing protocol

In the integrated approach, bandwidth is calculated in the RREQ phase in con-junction with route discovery. This way in the RREQ phase the protocol will try to nd a path with su cient bandwidth, not to check whether a path has su cient bandwidth *after* it has been found. Bandwidth can be calculated on its path as a RREQ packet is forwarded. When a RREQ reaches the destination via a path P, the bandwidth calculation on P is complete. Like the decoupled approach, to nd the available bandwidth on a path requires the calculation to be done all the way from end to end. This excludes any node other than the destination to generate a RREP. As a RREQ is forwarded hop by hop and leaves behind a path $F\ P$, the available bandwidth on $F\ P$ is calculated. If a node nds that $F\ P$ cannot meet the required bandwidth, it drops the RREQ. So this RREQ does not reach the destination and no RREP is generated for this path. This guarantees that a path found by the integrated approach has enough bandwidth to satisfy the QoS requirement.

When a source node wants to setup a QoS route for a flow to a destination, it sends a RREQ as it starts the route discovery. The RREQ carries the flow information. A partial path from the source, $F\ P$, is set up as the RREQ prop-agates from the source. The forward algorithm ($F\ A$) is used to calculate the bandwidth on the partial path $F\ P$ the RREQ has traversed so far. Transitions are shown in Figure 5.5 and the conditions and operations associated with each transition are defined below:

U: *N ON E ! REQ*: An entry for a QoS route is setup when the source of the flow sends a RREQ, or when a non-source node receives and forwards a RREQ, or when the destination receives a RREQ and veri ed there is su cient bandwidth on the route. The process of the RREQ has been described early. A node records the neighbor from which it receives the RREQ as its upstream neighbor on the route. The length of the timer is set to *Route setup time*. *REQ ! N ON E*: The entry for the QoS route is deleted when the timer expires and no QoS route is setup; *REQ ! RESV* : The state becomes *RESV* when the destination sends out a RREP, or a node on the route, including the source, receives a RREP.

For a node other than the source, it also updates the RREP packet and forwards it to the upstream neighbor. It records the neighbor from which it receives this RREP as its downstream neighbor on the route. The process of the RREP has been described early. The length of the timer is reset to *Route setup time*.

[39]. *RESV ! RESV* : The state *RESV* is refreshed when a data packet belonging to this flow is sent by the source, or forwarded by an intermediate node, or received by the destination. The timer is reset to *Route lif e time*. Once a QoS route is setup, it is used during the lifetime of the session, unless it breaks due to some topological change. In order not to disturb the packet flow, a QoS route is not changed as long as the required QoS is satis ed; *RESV ! BRK U* : The *RESV* state becomes *BRK U* when no data packet arrives for *Route lif e time* and the timer expires. This implies the QoS route is broken at the upstream. The timer is set to *Route setup time*.

[40]. *BRK U ! RESV* : The QoS route which was broken at upstream is restored. The timer is set to *Route setup time*. This could happen for three cases. The rst case, a data packet belonging to this flow arrives, indicating the QoS route from the source to the current node has been restored. The second case, a node n_k receives a RREQ packet from node $n_{k+1'}{}^{x}$. After calculating the bandwidth of the path $F\,P^{k0}$ along which this RREQ traveled from the source to itself, and verifying there is enough bandwidth on this path, it sends out a RREP back to $n_{k+1'}$, even it may not be the destination. Note that node $n_{k+1'}$ is not its upstream neighbor n_{k+1} on the original QoS route (n_{k+1} will reply, rather than forward the RREQ if it receives one). The state transits to *RESV* when this node sends the RREQ and the timer is set to *Route setup time*. If this node is the destination, this is identical to the initial route discovery phase. If this node is not the destination, this can be called a local reply. Note that in the initial route discovery phase, only the destination can send a reply. What makes the local reply feasible here is that the part of the original QoS

route from this node to the destination (BP^k) still exists, although most likely every downstream node is also at *BRK U* state. When the RREP reaches the source, a QoS route is setup between the source and the current node. This, together with the part of the original route from the current node to the destination, restores the entire route. Local reply reduces the delay to restore a broken route. A node sending a local reply also sends a route hold packet (RT HLD) towards the destination. On receiving the RT HLD, nodes at the downstream also transit to *RESV* (this is the third case), so the QoS route at the downstream side is reinstated.

A potential problem for allowing any *BRK U* node to locally reply the RREQ is that more than one routes can be built. This happens when more than one *BRK U* node send out local replies. Although these routes do not form a loop (they are all from the source to the destination), this is apparently redundant. Which route will be used depends on which RREP reaches the source rst. When a node in *BRK U* sends a local reply, it may temporarily have two upstream neighbors: the one it sends the local RREP to and the one on the original QoS route. The route from the original neighbor cannot be deleted at this moment, because one of its upstream neighbors could also send a reply (and assume the original down stream route is still good). This route may still be used. As data packets start to flow on one of the routes, they will refresh the *RESV* states on that particular route. Others routes will time out and be deleted. As a result, route redundancy is only temporary and there is only one QoS route per flow after the states stabilize. *BRK U ! N ON E*: The route is deleted at this node if it cannot be restored when the timer expires. The slots $T\ S_k^P$ are released; *RESV ! BRK D*: When a node nds the link to its downstream breaks, the route breaks and it transits to *BRK D*. At the same time it sends a route error packet (RERR) towards the source. A node also transits from

RESV to *BRK D* when it receives a RERR packet from its downstream neighbor. As the RERR packet is forwarded from the broken link towards the source, every node in this part of the route becomes *BRK D*. The timer is set to *Route setup time*.

R *BRK D ! REQ*: If this node is the source, it sends out a new RREQ as soon as it receives the RERR and transits to *REQ*. If this node is not the source, it becomes *REQ* when it receives (from $n_{k+1'}$) and forwards a RREQ packet. Suppose this node is n_k , and its upstream (downstream) neighbor on the original QoS route is n_{k+1} (n_{k-1}). The transmission slots on link (n_{k+1} *!* n_k) is $T\ S_k^P{}_{+1}$ and on link (n_k *!* n_{k-1}) is $T\ S_k^P$. It is possible that $n_{k+1'}$ and n_{k+1} are not the same. When processing the RREQ, node n_k uses

$$SRR_k^0 = SRR_k [\ T\ S_k^P{}_{+1}; \tag{5.35}$$

$$SRT_k^0 = SRT_k [\ T\ S_k^P \tag{5.36}$$

in the place of SRR_k and SRT_k . Although slots $T\ S_k^P{}_{+1}$ and $T\ S_k^P$ are reserved on the old route, they can be used on the new route as well. The timer is set to *Route setup time*; *BRK D ! N ON E*: The QoS route entry is deleted if no RREQ arrives before the timer expires. The slots $T\ S_k^P$ are released. *RESV ! N ON E*: When transmission of the session is complete and the QoS route is not needed anymore, the source node sends a route release packet (RT RLS) to release the route. Transmission time slots $T\ S^P$ on P are released when the RT RLS reaches the destination.

The parameters *Route setup time* and *Route lif e time* should reflect the dynamics of the QoS routing protocol. The timer is set to *Route setup time* for route discovery and route repair. It should be long enough for a packet to be transmitted back and forth on the route. *Route lif e time* should be in the order of data packet arrival interval, because on an established route data packets flow regularly and the timer is refreshed by every packet. This allows quick detection once the route breaks and the data packet flow stops. Because soft-states are used and transitions can be triggered by timers, under no circumstances does a node keeps a route forever. Eventually all states become *N ON E*, the QoS route is deleted and the time slots are released.

An example of route setup and route repair with the QoS routing protocol. The direction of a packet is shown with the arrow by transmitting a RREQ. The RREQ packet is forwarded throughout the entire network (Figure 5.6.a). For simplicity, we assume there is enough bandwidth on every link so the RREQ packet is not dropped. On receiving and forwarding the RREQ, every node sets up an entry for the route and sets the associated soft-state to *REQ* (Transition 1). When the RREQ reaches the destination n_0 via a path $P = fn_4 \,!\, n_3 \,!\, n_2 \,!\, n_1 \,!\, n_0 g$, n_0 sends a RREP to n_4 in the opposite direction of P (Figure 5.6.b). The state at n_0 becomes *RESV* . On receiving RREP, nodes on P determines and reserves transmission slots $T\ S^P$ Their states transit to *RESV* (Transition 3). A QoS route P is established. As data packets sent by n_4 travel along P , the *RESV* states of the nodes onare refreshed periodically (Transition 4). For a node not on P (n_5, n_6), the route entry is deleted (Transition 2) when no RREP packet is received before the timer expires. Suppose sometime later a node n_1 on P moves from the vicinity of n_2 to the vicinity of n_6. The link between n_1 and n_2 breaks and a new link appears between n_1 and n_6. Assume the link between n_1 and n_0 is not a ected by this movement. The node upstream of the broken link (n_2) detects its next hop node (n_1) is gone and sends a RERR packet back to the source (Figure 5.6.c). Nodes n_2, n_3 and n_4 become *BRK D* (Transition 9). In the meanwhile, nodes downstream of the broken link (n_1, n_0) time out when they do not receive data packets of the flow for *Route lif e time* and transit to *BRK U* (Transition 5). When the source node n_4 receives the RERR packet, it sends out a new RREQ and starts a new round of route discovery (Figure 5.6.d). Every node which either does not have an entry for the QoS route (n_5, n_6), or where the route state is *BRK D* (n_3, n_2) receives and forwards the RREQ. Their states become *REQ* (Transition 1 for n_5, n_6 and Transition 9 for n_4, n_3 and n_2). When the RREQ reaches n_1 via $F\ P^{\,0} = fn_4 \,!\, n_5 \,!\, n_6 \,!\, n_1 g$, if the soft-state *BRK U* at n_1 has not expired, n_1 generates a local reply and sends out the RREP back to the source in the reverse direction of $F\ P^{\,0}$ (Figure 5.6.e). The

state at n_1 becomes *RESV* (Transition 6). At the same time n_1 sends a route hold packet (RT HLD) to its downstream neighbor n_0. Node n_0 also becomes *RESV* (Transition 6). As the RREP is forwarded back to n_4, every node on $F\ P^{\ 0}$ (n_6, n_5, n_4) determines and reserves their transmission time slots. Their states become *RESV* (Transition 3). The route is restored whenthe RREP arrives at n_4. The soft-states at nodes n_2, n_3 time out and their route entries are deleted (Transition 2). As data packets flow through this new route $fn_4\ !\ n_5\ !\ n_6\ !\ n_1\ !\ n_0g$ (Figure 5.6.f), the *RESV* state at every node on the route is being refreshed periodically (Transition 4).

5.6 Simulations results

The performance of the QoS routing protocol is studied with simulations. The QoS routing protocol has been implemented with $N\ S\ -\ 2$. The implementation is based on the AODV module contributed by the MONARCH group, and the QoS routing functions are added. In additional to QoS routes, a node also stores a best-e ort route in its routing table when it learns such a route. A best-e ort route is used when a QoS route is not available. In the simulations, *Rount setup time* = 1000 ms and *Route lif e time* = 200 ms. E-TDMA is used at the MAC layer for all the simulations. The parameters of E-TDMA are the same as those used in Chapter 3 and are not repeated

The setup the simulations is same as Chapter 3. Networks of 25 nodes and 40 nodes are generated, where nodes roam in an area of 1000m by 1000m, or 1250m by 1250m, respectively. The network is always connected, i.e., network partition is not allowed. Network mobility is varied by changing the maximal nodal speed v. We use v = 0,5,10 m/s to model di erent mobility. At v = 10 m/s, on average a node has a link change every 5 seconds. Network tra c is generated by CBR source, where the source and the destination of a session are chosen randomly. A CBR source generates packets of 64 bytes (a packet becomes 84 bytes after IP header is added) at a rate of 20 packets per second, and a session lasts 30 seconds. The network load is varied by changing the number of CBRsessions generated

during the simulations. A source always transmits its 600 packets disregarding how many of them get through. (No admission control is used, because it is not possible with the original best-e ort AODV protocol, which is used for comparison.) If a QoS route can be setup for a session, the packets are transmitted on the QoS route; otherwise they are transmitted on a best-e ort route. The duration of the simulation is 300 seconds. We compare the simulation results from the QoS routing protocol and the original, best-e ort AODV protocol (BE). The two protocols will be compared at both the packet level (packet throughput and average delay) and the session level (session good-put and average packet delay of serviced session). As a crude measurement of the service received by a session, a session is called "serviced" if at least 90% of its packets reach the destination. The simulation results for the smaller networks are presented rst. These results can be found is from Figures 5.7 to 5.12.

Figures 5.7 and 5.8 show the packet throughput and the average packet delay under di erent tra c loads and node speeds. Under light tra c, packet through-put and packet delay are very close for the two protocols, because they often use same routes. The advantage of QoS routing protocol becomes apparent when tra c gets heavy. With the BE protocol, a node has at most one active route to a destination. It uses this route to transmit or forward all the packets to this destination, irrespective of the congestion on this route. As the network tra c becomes heavy, the single route used by the BE routing protocol becomes heavily loaded, causing packets to be delayed and dropped. The average packet delay increases signi cantly under heavy tra c. On the other hand, the QoS routing protocol tries to nd and use routes satisfying bandwidth constraints for di erent flows, even between the same pair of source and destination. Two QoS routesmay share the same path, but the protocol will ensure enough bandwidths are reserved on this path to accommodate both flows. The tra c load is more balanced this way. The average packet delay increases with o ered load slowly with the QoS routing protocol. In fact the average packet delay of serviced sessions is much lower than the average delay of all the sessions (Figure 5.11). Packets from sessions not serviced, often sent over best-e ort routes, contribute to much of the delay, especially under heavy trafic. When the nodal speed v increases, the throughput of both protocols drops. Mobility affects network

throughput at both the MAC layer and the routing layer. At the MAC layer, it takes time for E-TDMA to resolve the collisions caused by node movement and to reserve new slots. Essentially a protocol like E-TDMA which is based on establishing reservation has only limited capability to handle network mobility and is best for a static network. At the network layer, it takes time for the routing protocol to re-establish a route when it breaks. While the source node of a flow may queue its packets while waiting for a new route, other nodes simply drop packets for a destination to which they do not have a route. For the QoS routing protocol, the packet throughput drops roughly by 15% at v=5 m/s and by 30% at v=10 m/s, compared with v = 0. Nodal mobility also increases the average packet delay. The average packet delay nearly doubles at v=10 m/s. Interestingly, when we compare the two routing protocols under mobility, the advantage of the QoS routing protocol increases. A possible explanation is as follows: because the QoS routing protocol uses di erent QoS routes for individual flows, when one of the QoS routes breaks, only this QoS route is repaired. Other are not a ected. Packets of the flow on the broken route are temporarily forwarded using the best-e ort route, which may coincide with one of the other QoS routes. Thereis more route redundancy with QoS routing. In the BE protocol, when the only route to a destination breaks, all packets addressed to this destination are de-layed or dropped. It can be expected that a best-e ort routing protocol which nds multiple routes will be better than AODV in this aspect. When the two protocols are compared at the session level (Figures 5.10 to 5.12), in the static network (v=0) both can service almost all the sessions up to 150 sessions. After that the BE protocol degrades until the session good-put drops to about 100. On the other hand, the QoS routing protocol continues to service more sessions. Average packet delay for serviced sessions is stable in both protocols. Note that the relative performance of the two protocols in terms of session good-put is very di erent from that of packet-throughput. With the BE protocol, all the packets are treated alike and transmitted in the order of arrival. Packets from di erent sessions are equally vulnerable to being dropped. When more sessions are transmitted at the same time, packets are dropped from all of them and fewer sessions deliver 90% of their packets. With the QoS routing protocol, it is possible to distinguish

packets from di erent sessions. Priority can be given to a packet transmitted on its QoS route before a packet transmitted on a best-e ort routed. With the QoS routing protocol the capacity reaches about 200 sessions. When nodes start to move, the session good-put for both protocols decreases signi cantly. Figure 5.12 shows that the probability for a session not serviced increases with the nodal speed v. For the QoS routing protocol, session good-put drops to 1/2 and 1/3 for v = 5 and 10 m/s respectively compared with v = 0. Once a route breaks, before it can be restored, the flow su ers signi cant degradation. The QoS routing protocol o ers little protection when this happens. Because of the bandwidth constraint, a QoS route is not alwaysrestored. (In contrast, a best-e ort route is usually restored.) In [67, 68], it was reported that the forced termination probability increases with more frequent topology change. The results here agree with their observations. We now look at the simulation results of the network with 40 nodes. These results are shown in Figures 5.13 to 5.18. We focus on where these results di er from those in the smaller networks. The average nodal density of the two networks are the same, but the average distance between a source and its destination is longer in the larger network. This is reflected in the higher packet hop count (Figure 5.15) in the larger network. When nodes do not move, the overall capacity of the larger network is higher because more packets are delivered and more sessions are serviced. This changes when the nodes start to move. By comparing Figure 5.7 with 5.13, and Figure 5.10 with 5.16, we can see that both the packet throughput and the session good-put decrease with node speed v more rapidly in the larger network. This is because in the larger network, a packet needs to travel more hops to reach its destination, and the probability that its route breaks due to nodal movement increases. The packet is more likely to be dropped. A longer QoS route is more di cult to establish and more di cult to repair than a shorter one. However, the QoS routing protocol still outperforms the BE protocol. For all these cases, the average packet delay for serviced sessions is less than 180 ms which can be tolerated by many real-time applications.

5.7 Discussions of the QoS and BE protocols

Mobility well, but no QoS is taken into account. When nodes move very fast, topology could change so quickly that one is lucky to nd a route at all, no to mention any QoS. Whether QoS can be achieved in a highly mobile network is questionable. At each node, there is at most one route to any given destination, and this route is changed when a fresher route, or sometimes a shorter route, is known. All the packets addressed to that destination are sent through this route, causing congestion on this route under heavy tra c. This leads to \hot spot" in the network where packets are delayed and dropped.

The QoS routing protocol builds individual QoS routes for di erent flows, even between the same source and destination. Packets transmitted on QoS routes are guaranteed of bandwidth. When an area of the network is congested, a new QoS route is likely to be built around it rather than through it, providing a form of load balancing. However, a RREQ to set up a QoS route has to reach the destination before it can be replied. A RREQ sent for a QoS route often travels further than a RREQ sent for a best-e ort route. In the worst case a QoS RREQ is flooded in the entire network, generating more overhead than a BE RREQ. Because of the requirement for bandwidth reservation, a QoS route is harder to construct than a best-e ort route. Failure of a link in the middle of a route will trigger rebuilding of the QoS route, which will involve every node from end to end. (On the other side, it is feasible to localize the e ort to repair a broken link in best e ort routing protocols.) The longer a QoS route, the more likely it breaks, and the higher cost for rebuild. As nodes move faster and the network topology changes more frequently, it becomes more and more di cult to build and to maintain QoS routes. All these suggest that the QoS routing protocol is only good for short routes and in networks of low mobility. Consequently QoS routes should be built and used as complement to, not substitute for, best-e ort routes.

Another advantage of the QoS routing protocol is related to the E-TDMA protocol used at the MAC layer. Route changes are more frequent in the original AODV protocol. Frequent route change requires frequent slot reservations and puts a heavier burden on E-TDMA. On the other hand, a QoS route is more stable. Once it is established, it does not change as long as it is not broken. Stable routes requires less frequent slot reservations and is better for E-

TDMA. However, these are characteristic of E-TDMA and may not be true if other protocols are used.

It should be recognized that although the QoS routing protocol performs signi cantly better than the BE protocol in the simulations, the ability for mobile ad hoc networks to provide QoS is very primitive compared with other types of networks. This ability is not only limited by the capacity of the wireless channel, but also adversely affected by the volatility (nodal mobility) inherent to these networks. The effect of mobility is clearly demonstrated in the simulation. Movement of a node can have a more sinister, sometimes catastrophic e ect in an ad hoc network than in a single hop wireless network such as a cellular network. A link failure will break every route going through it and affects many nodes, while in a cellular network breakage of a link only affects that particular user. If the nodes move very fast, the relative delay for the protocol to establish end-to-end QoS routes becomes too long and cannot keep up with the topology change. In that case the protocol becomes inapplicable and inferior to a BE protocol designed to handle topology changes quickly.

A major criticism of this QoS routing protocol is that it is designed without considering the situation when multiple QoS routes are being setup simultane-ously. A route request is processed under the assumption that it is the only one in the network at the moment. When multiple routes are being setup simulta-neously, they each reserve their own transmission time slots. When they cross, they may compete for the same set of slots and interfere with one another. It is possible that two QoS routes will block each other when they are trying to reserve the same time slots simultaneously; but if the two requests come one after another, one of them will be successful. This is because no attempt is made to coordinate di erent route requests. This is not a problem for the BE protocol, where no resource reservation is necessary and two routes can simply cross each other. However, the use of soft-states ensures there will not be dead-locks between the two competing QoS routes. If two QoS routes cannot be fully established because they are blocking each other, both will be deleted. Howto setup QoS routes when there are multiple competing requests needs further study.

5.8 Conclusion

An on-demand QoS routing protocol based on AODV is developed for TDMA-based mobile ad hoc networks. Upon request of the higher layer, it can build a QoS route from a source to a destination with reserved bandwidth. It is designed for sessions which transmit with constant bit rate. The bandwidth calculation problem associated with QoS routing in these networks is also studied. We showed to calculate the maximum available bandwidth on a route is NP-complete and designed an e cient distributed algorithm. This bandwidth calculation algorithm is integrated into the AODV protocol in search of routes satisfying the bandwidth requirements. Besides nding a QoS route, the QoS routing protocol can also restore a route when it breaks due to some topological change. Therefore it can handle some degree of network mobility. Its performance is compared with that of the original AODV protocol with simulations. The simulation results show that the QoS routing protocol can produce higher throughput and lower delay than the best-e ort protocol. It works the best in small networks or short routes under low network mobility.

Chapter 6

Conclusions

Medium access control and quality-of-service routing for mobile ad hoc networks deploying TDMA have been studied. The problem of generating TDMA trans-mission schedules is studied in Chapter Two and Chapter Three. A Five-Phase Reservation Protocol for broadcast

scheduling has been developed. A new ve-phase message exchange mechanism allows a node to reserve a broadcast slot by only interacting with its neighbors. Instead of waiting in turn to reserve their slots, in the FPRP protocol nodes use contention to acquire their slots. The ve-phase conversation resolves conflicts arisen from contentions. It does not su er from the hidden node problem encountered in some other contention-based pro-tocols. The protocol is concurrent in the sense that it runs all over the network at the same time and many nodes can reserve their slots simultaneously. The quality of the schedule generated by FPRP is comparable to that of a schedule generated by a greedy scheme. Rivest's pseudo-Bayesian algorithm is modified to work with FPRP in the multihop environment. It is suitable for large mobile networks.

In Chapter Three we have developed an Evolutionary-TDMA scheduling pro-tocol for generating schedules including unicast, multicast and broadcast trans-missions, and for updating these schedules as the network topology and band-width requirement change. It uses FPRP as its signaling scheme, so it inherits the concurrency of FPRP. It too is suitable to use in a large mobile network. As the network topology and tra c pattern change, a node can adjust its schedules, release time slots for completed or corrupted transmissions and reserve new slots as needed. By transmitting in reserved, nearly conflict-free time slots, a node can better guarantee the QoS of its tra c. This is important for tra c with stringent bandwidth or delay requirement, especially under heavy network load.

Power control and channel probing has been studied in Chapter Four. The widely used closed-loop power control algorithm is adopted. Our contribution is the developement of a new channel probing scheme which works in a wireless network employing the power control algorithm. When a link probes a channel by simply transmitting a probing signal and measuring the power and the SIR, it gains useful information of the channel regarding its feasibility to achieve its target SIR and the required transmission power. This scheme di ers from other channel probing schemes in that it allows multiple links to probe a same channel simultaneously in a distributed and uncooperative manner. The equivalence between the local admissibilities of these links obtained from channel probing and the global feasibility of all

the links in the channel has been proven. For a TDMA or FDMA system, information obtained from channel probing can improve the performance of dynamic channel allocation; for a DS/CDMA system using conventional matched lter receiver, channel probing can be used as an admission scheme and better protect admitted calls from newly arrived calls.

Quality-of-Service routing has been studied in Chapter Five. We developed a QoS routing protocol based on AODV for TDMA based mobile ad hoc network. This protocol can setup bandwidth reserved routes for sessions transmitting at constant rate. We studied the bandwidth calculation problem for TDMA net-works and found that it is NP-complete to nd the maximum available end-to-end bandwidth. Following this discovery, we developed an e cient distributed algorithm to calculate the path bandwidth, either in the forward direction or in the backward direction. The performance of this algorithm was compared with an upper-bound of the end-to-end bandwidth and was found reasonably good. After briefly discussing a decoupled approach which uses bandwidth calculation for admission-control, we described in detail the integrated QoS routing proto-col which incorporates bandwidth calculation with the AODV route discovery mechanism to nd routes satisfying the bandwidth requirement. Soft-states are used to protect a QoS route. The protocol can restore a QoS route when it breaks due to some topological change. Together with the E-TDMA protocol used at the MAC layer, the QoS routing protocol provides a solution to support QoS in small networks with relatively low mobility. Simulations showed that it achieves higher throughput and lower delay than the original AODV protocol.

To summarize, di erent aspects of mobile ad hoc networks have been inves-tigated in this dissertation. However, di erent networks can be dramatically di erent in size, mobility, communication and computation capability, require-ment and energy constraint. There is no \one size t all" solution. In particular, the scheduling protocols and the QoS routing protocol are developed to provide QoS in networks of relatively low mobility. They do so by setting up states for individual flows, which comes at a non-negligible cost. Consequently they fall into the category of InteServ [76]. It has been recognized that such an approach has limited scalability and limited capability to handle network changes. An al-ternative is to classify

packets into a number of di erentiated service classes and serve them on a per-class/per-hop, not per-flow/end-to-end basis. The Di Serv model [77], though relatively simple and coarse, o ers more flexibility and may better handle the inherent dynamics in mobile ad hoc networks. Use of Di Serv model in mobile ad hoc network has not been widely studied, but it deserves an in-depth investigation. It will be very interesting to compare these two di erent approaches.

