TECHNICAL RESEARCH REPORT

An Evolutionary-TDMA Scheduling Protocol (E-TDMA) for Mobile Ad Hoc Networks

by Chenxi Zhu, M. Scott Corson

CSHCN TR 98-14
(ISR TR 98-32)

An Evolutionary-TDMA Scheduling Protocol for Mobile Ad Hoc Networks

Chenxi Zhu and M. Scott Corson
Institute for Systems Research, University of Maryland
College Park, Maryland 20742
(czhu,corson@isr.umd.edu)

Abstract— **A new single channel, time division multiple access (TDMA) scheduling protocol, termed "Evolutionary-TDMA", is presented for mobile ad hoc networks. The protocol allows nodes in an ad hoc network to reserve conflict-free TDMA slots for transmission to their neighbors. Two topology-dependent schedules are generated and maintained by the protocol: a broadcast schedule suitable for network control traffic and a mixed schedule which combines unicast, multicast and broadcast transmissions for user data traffic. The schedules are frequently updated in an evolutionary manner to maintain conflict-free transmissions. The protocol executes across the entire network simultaneously in a fully-distributed and parallel fashion. Traffic prioritization and Quality of Service (QoS) can be supported. Simulations have shown that the performance of the E-TDMA protocol is close to that of centralized algorithms, while being insensitive to network size in terms of scheduling quality and scheduling overhead. It is a scalable protocol suitable for very large networks, and networks of varying size.**

I. Introduction

Mobile ad hoc networks (a.k.a. mobile packet radio or mobile, multihop wireless networks) have gained much attention in recent years. An ad hoc network consists of a number of geographically-distributed, potentially mobile nodes sharing a common radio channel. Due to its self-configurable nature, an ad hoc network can be rapidly deployed in an area without the aid of a fixed infrastructure. The multihop topology of an ad hoc network allows spatial reuse of the wireless spectrum. Existing commercial standards [1], [2] for ad hoc networks only employ contention-based medium access control (MAC) schemes. However, much work has been done on reservation-based MAC approaches based on techniques such as time division multiple access (TDMA). Reservation of TDMA slots, or "TDMA scheduling", refers to the problem of assigning the TDMA transmission slots among the nodes in a way which both avoids conflict and allows efficient spectrum reuse.

Many algorithms have been developed for TDMA scheduling in ad hoc networks (cf. [3], [4], [5], [6], [7], [8], [9], [10], [11], [12], [13], [14], [15], [16]). The broadcast scheduling problem is addressed in [3], [7], [8], [11], [16] and the unicast scheduling problem is addressed in [5], [8], [9], [10]. Some of these algorithms are centralized algorithms requiring a central controller with the knowledge of the entire network. In other protocols [4], [11], [10], the topology information is obtained gradually, but an "initiator" node is needed to initialize the scheduling process. Furthermore, some of these protocols depend on the existence of such an initiator node *throughout* their execution, and thus are less desirable from the perspective of robustness and survivability. Some protocols use a fixed TDMA schedule to aid in slot assignment, where every node is preassigned a slot [7], [8], [9]. However, this approach has limited scalability. Others use a cluster structure partially emulating the cellular concept, where an elected clusterhead performs the necessary coordination and control—including TDMA scheduling—in a neighborhood (cluster) [5], [15]. The approach of some recent work is to generate topology-independent schedules [13]. Finally, the work of [3], [6] specifically addresses the issue of re-scheduling when the network changes.

The subject of this paper is an algorithm which generates and maintains TDMA transmission schedules which accommodate both a randomly-changing topology and dynamic bandwidth requirements in a fully-distributed, parallel and evolutionary fashion. The rest of the paper is organized as follows: Section II puts forth the considerations that shaped the algorithm's design; Section III describes the algorithm; Section IV provides a detailed example of its operation, and Section V evaluates the performance of the protocol via simulation. Section VI discusses various characteristics and applications of the protocol; and Section VII provides some concluding remarks.

II. Design Considerations for Transmission Scheduling

The radio channel readily supports broadcast communications. When a node transmits using an omni-directional antenna, every other node within its transmission range receives its packet. If we do not consider the capture effect, from a transmitter's point of view, when it is transmitting a packet to a one-hop neighbor, it is blocking all the other neighbors from receiving from other sources. From a receiver's perspective, to receive a packet successfully prohibits all its one-hop neighbors, except the intended transmitter, from transmitting. Thus, a node cannot transmit and receive simultaneously (no primary interference), and it cannot receive more than one packet at a time (no secondary interference). A transmission is successful only if the packet is the only one received by the receiver, and the receiver itself is not transmitting at the same time. Scheduling in a network like this is difficult, because nodes as far as two hops apart can conflict, but cannot communicate directly with each other. There are three classes of transmissions: unicast, multicast and broadcast, designating delivery to one, some or all of the one-hop neighbors of the transmitter, respectively. 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 and with all the neighbors as receivers). The transmission requirement found in a real ad hoc network is often a mixture of unicast, multicast and broadcast, where the majority of the data traffic will likely be unicast and multicast—with broadcast typically being used for network control and organization activities. While the traffic generated by network control and management is roughly uniform when distributed algorithms are used, the user traffic can be highly irregular. The amount of bandwidth required by different nodes can vary dramatically. A node should be able to reserve different amounts of bandwidth, possibly using different transmission types. When the network is congested and there is not enough bandwidth to satisfy all the requirements, some traffic should be given higher priority than others. Generally speaking, traffic for network control should have higher priority than user-generated traffic, and real-time traffic (such as voice and video) should have higher priority than non-real-time traffic. A scheduling policy should also be fair to all nodes, and no node should be starved of transmission bandwidth.

In the parlance of graph theory, transmission scheduling in an ad hoc network 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 traffic is a mixture of node coloring and edge coloring. To produce the optimal schedule (where optimality is measured in terms of bandwidth efficiency; i.e. we desire schedules with the *minimum* number of TDMA slots) is an NP-complete problem for an arbitrary network [17], [7] and is intractable. However, the most bandwidth-efficient schedule might not be the best for a mobile ad hoc network, because it has the *least* redundancy (i.e. it has the highest spectral reuse factor) and is therefore the most susceptible to being corrupted by topological change. 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 first generated. The schedules also need to accommodate changes in bandwidth requirements. As old transmission sessions end and new sessions begin, bandwidth should be released from terminated sessions and assigned to new sessions quickly. All these changes, both in network topology and in network traffic, force the transmission schedule to be updated. This is referred to as schedule "maintenance". Because maintenance has to be performed frequently to accommodate these changes, it has to be done in a cost-effective manner. Compared with other types of networks, an ad hoc network is often limited both in bandwidth and in computation power. It is desirable that the communication and computation overheads required to generate and to maintain the transmission schedules be as low as possible. A brute force approach, which tears down the existing schedule completely whenever changes occur in the network and regenerates a new one, is apparently inappropriate. Although a new schedule reflects the latest network topology and bandwidth requirements and can be made very efficient, its generation is likely too costly and somewhat redundant, especially when only a small part of the existing schedule is outdated and the rest is still valid. A more natural solution is an "evolutionary" approach. In this approach, the existing schedules are kept as much as possible. Only the part which is outdated, either due to node mobility or due to changing bandwidth requirements, is updated. If the interval between two updates is short enough, only a small portion of the existing schedule needs to be updated. Compared with regenerating the entire schedule, this method is more economical. Once a flow has had its bandwidth reserved, it will have its exclusive use until the flow ends (in which case the bandwidth is released) or when the reserved bandwidth becomes corrupted (in which case the network control protocol will attempt to reserve some new bandwidth for the on-going flow). Quality of Service (QoS) support in ad hoc networks is more feasible with this approach.

Due to the dynamic nature of an ad hoc network, distributed protocols are preferred over centralized protocols. This is important both for efficiency purposes and for robustness and survivability. Nodes may malfunction or be destroyed, and it is desirable that the scheduling process not depend on a particular node. A real network could be extremely dynamic, both in size and in topology. It 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. Nodes should be allowed to join the network dynamically. The protocol needs to handle different network topologies (ranging from sparsely to densely-connected), and when its performance degrades, it does so gracefully. An efficient schedule is necessarily *topology-dependent*. While collecting more topology information results in a better schedule, more overhead is also required. Because a transmission from a node only affects its neighborhood and not other remote nodes, the scheduling process can be kept *local*, especially when the goal is not to generate an optimal schedule. This is particularly important when the network is large, and the schedule needs to be updated quickly.

The preceding highlights what we consider to be important characteristics for a scheduling protocol. Here our intention is not to produce the most bandwidth-efficient schedule, but to produce and to maintain a conflict-free schedule as rapidly as possible in a fully-distributed, parallel fashion with only local knowledge. The design of the protocol incorporates almost all of these characteristics, falling short principally in the protocol's inability to gracefully handle large variations in nodal degree (more on this later).

III. The Evolutionary-TDMA Protocol

The Evolutionary-TDMA (E-TDMA) protocol allows nodes in an ad hoc network to assign TDMA transmission slots among themselves as network composition and bandwidth demands *change*. The protocol produces two TDMA schedules *simultaneously*. The first schedule is a broadcast (i.e. node) schedule, in which every node is assigned one slot. This broadcast schedule can be used for traffic generated by network control protocols, including the E-TDMA protocol itself. The second schedule carries user generated traffic and can be very flexible. It can be a *mixture* of unicast, multicast and broadcast transmissions. The reservations performed here are *one-hop* reservations. A node can schedule different amounts of bandwidth to transmit to one, or some, or all of its neighbors freely, and to reserve different number of transmission slots depending on its need. Both schedules reflect the topology of the network and are conflict-free. Furthermore, 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 quickly—in an evolutionary manner—in order to minimize the rescheduling 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 schedules are generated in a fully-distributed and parallel fashion. The scheduling process is executed across the *entire network* at the *same* time. This reduces the scheduling overhead and enhances the robustness and survivability of the network. Essentially, every node is responsible for its own transmission schedule. A node only needs to communicate with its one-hop neighbors in order to produce conflict-free schedules. A greedy algorithm is employed to make the schedule efficient, given the limited, local knowledge present at each node. It supports dynamic membership, i.e., a node can join the network and participate in the protocol at any time. The protocol's performance, in terms of both the quality of the generated schedules and the scheduling overhead, is insensitive to network size. It is especially 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 bandwidth-symmetric and bidirectional. The topology of the network can be represented by an undirectional graph $G(V, E)$, where V is a set of nodes and E is a set of edges;
- The network topology changes *slowly* relative to packet transmission time. During the interval when the schedule is being updated, we assume the topology does not change.
- Every node is able to operate the Five Phase Reservation Protocol (FPRP) [16].

A. Overview of the Protocol

The protocol operates within a single TDMA channel. The channel is partitioned into two portions: a control and organization portion, where TDMA schedules are periodically generated and updated, and an information transmission portion, where the TDMA schedules are used to transmit information packets. The control and organization portion is called the Control/Organization Frame (COF). Every COF is followed by m Information Frames (IF), each of which consists of n Information Slots (IS) (Figure 1). In each information frame, a node transmits or receives information packets according to its schedule, called the Information Frame Schedule (IFS). The IFS of an information frame is produced in the preceding COF. The COF oper-

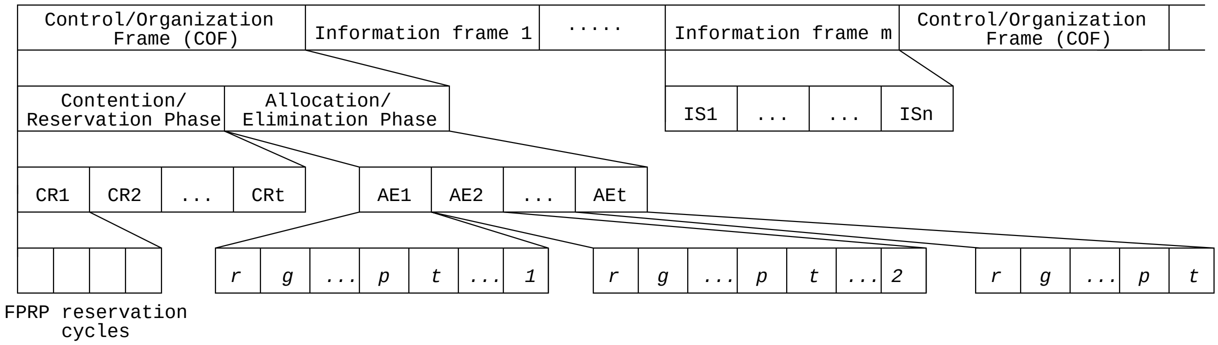


Fig. 1. Structure of the E-TDMA protocol. The permanent colors are $r, g, ..., p$ and the temporary colors are $t, ..., 1$.

ates with its own schedule, called the Control/Organization frame schedule (CFS), which is updated at the beginning of the COF.

The E-TDMA protocol operates in the COF. It generates and maintains the CFS and IFS in the face of a changing network topology and changing bandwidth requirements. Because the nodes need to cooperate extensively in the control phase, the CFS is a broadcast schedule, and every node is assigned a slot. In graph theoretic terms, assigning every node a conflict-free slot is equivalent to assigning every node a color, with the constraint that no two nodes two hops apart or less are given the same color. For this reason, a slot in the CFS is called a "color". A node broadcasts in the COF slots assigned to its color. In the IFS, different nodes have different requirements for transmission bandwidth, and these requirements can be of different types. The IFS is a mixture of unicast, multicast and broadcast schedules, and an IS in IFS is simply called a "slot". Information packet traffic—which can be voice, video, data, etc.—is represented by "flows". In a particular slot, the schedule of a node i can be in one of the following states: transmitting to a neighboring node j, labeled as $T(j)$; receiving from node j, $R(j)$; blocked-from-transmitting because a neighboring node j is receiving a packet from another node, $Bt(j)$; blocked-from-receiving because a neighboring node j is transmitting to another node, $Br(j)$; simultaneously blocked-from-transmitting due to node j and blocked-from-receiving due to node k, $Btr(j;k)$; idle, I, when it is not in any of the above states. If a node is blocked-from-receiving by packets coming from unknown source (e.g., in the case of a collision), this slot is labeled $Br(X)$. The states Br and Bt are mutually exclusive, unless explicitly specified as Btr. Due to the transmitting-or-receiving constraint, a T node is also a Br node, and a R node is also a Bt node, but these are still referred to as T or R, respectively. The transitions among these states should be clear from their definitions. A node is ready-to-transmit (RTT) if it is either idle (I) or only blocked-from-receiving (Br), and is ready-to-receive (RTR) if it is either idle (I) or only blocked-from-transmitting (Br).

When nodes move around, the topology of the network changes and collisions occur in the schedules. Both the CFS and the IFS may be corrupted. The corrupted portions need to be fixed in the next COF. Since the last COF, some flows may have ended and some new flows may have arisen. These changes in bandwidth requirements need to be addressed as well. In order to execute the scheduling protocol in the COF, first the CFS needs to be brought to a working condition. The IFS is then updated using the CFS. As an evolutionary protocol, the schedules are updated on the basis of the existing parts. The main challenge to be met is to let the nodes update their schedules while ensuring that the new assignments do not conflict with the existing schedules and with each other. The following observations are key to the solution:

Observation 1: If a node knows the up-to-date schedules of all its one-hop neighbors, it is able to pick a conflict-free slot to transmit to one (or some, or all) of them. It can pick a slot in which itself is RTT and the receiver(s) is RTR. This transmission will not interfere with others.

Observation 2: If two nodes are at least three hops away, their schedules will not interfere with each other, either directly or indirectly. As a consequence, two nodes three hops apart or further can schedule their transmissions independently. No collision will occur.

The scheduling protocol is "transmitter-oriented"; i.e., the transmitting node, or the sender, is fully responsible for the reservation, maintenance, and release of the transmission slot. When a new flow arises, the transmitter reserves a slot to transmit to the receiver. When a flow ends, the transmitter releases the slot. If a transmission is interrupted, either because of a link failure or because of a topological changes, the transmitter is required to detect (i.e. to be informed of) this promptly[1]. It is also the

[1]The mechanism for detecting such link failures or conflicts is not part of this protocol, but should be provided by a lower or upper layer

transmitter's responsibility to release the current slot and to reserve a new one for the on-going flow. The receiver never solicits a transmission from the sender, either when a flow starts or when a flow is interrupted. It releases a slot when the sender does not claim it any more. Multiple nodes can make their reservations for transmission slots simultaneously, as long as they do not conflict. Within a neighborhood where nodes interact with each other, the reservation procedures of different nodes are interleaved so that no conflict can occur. This guarantees the schedules are always free from collision.

B. *Detailed Description of the Protocol*

In order to illustrate the evolutionary nature of the protocol, we will describe how it works in a scenario where a set of existing schedules (the CFS and IFS) have already been running in the network. It will be seen later that network initialization (where old schedules are non-existent) is a trivial extension of this scenario. The COF is responsible for updating both the CFS and the IFS. It has two phases: a Contention/Reservation phase followed by an Allocation/Elimination phase.

B.1 Contention/Reservation Phase

The purpose of the Contention/Reservation (CR) phase is to update the CFS. It allows every node to have a conflict-free broadcast slot (a "color") in the CFS. When a CR phase begins, every pre-existing node has a color assigned to it in the CFS from previous COF. This color is called a "permanent color". How these permanent colors are assigned will be clear later. Due to topology change, some nodes may find their permanent colors corrupted. These nodes need to obtain new colors. If we assume that the interval between two COFs is short relative to intervals between corruptions caused by node mobility (meaning that most of the permanent colors are still valid), only a small fraction of the nodes will have invalid permanent colors and will need to acquire new ones. Meanwhile, some nodes may wish to obtain new transmission slots in the IFS, either for newly arrived (or backlogged) flows, or for on-going flows corrupted by some topological change. These two groups are not exclusive: if a topological change corrupts the permanent color of a node, it may corrupt its transmission slots as well. The CR phase assigns these nodes "temporary color"—which are valid only for the current COF—and updates the CFS *temporarily*. There are t temporary colors. The number of temporary colors is chosen to reflect the network requirements so that the contending nodes are successful with high probabilities. Both the permanent colors and the temporary colors are used in the next phase, as will be seen shortly. The temporary colors are labeled as "1,2,...,t", where color 1 represents the highest priority and color t the lowest. Nodes that do not contend for temporary colors participate in the current COF using their permanent colors assigned earlier. These permanent colors are labeled as "$r(ed), g(reen), b(lue), y(ellow)$," etc. There are p permanent colors, and this number is chosen to reflect the expected local connectivities of the network. There are more permanent colors than temporary colors, since only a small fraction of the nodes need updating (and thus need temporary colors), while all the nodes need permanent colors in order to participate in the protocol.

A CR phase consists of t Contention/Reservation slots (CR1,...,CRt), each corresponding to a temporary color. Nodes that need to obtain new colors contend for these temporary colors. The Five Phase Reservation Protocol [16], which allows nodes to reserve conflict-free broadcast slots, is used during the contention. During the CR slots, nodes simultaneously *contend* for the temporary colors and *reserve* them for future use with the FPRP protocol. Thus there is a mixture of contention and reservation in this phase (hence the name). The internal structure of each CR slot is defined by the FPRP protocol, each consisting of a number of reservation cycles. There should be enough reservation cycles to allow each temporary color to be assigned thoroughly (see [16] for details of the FPRP protocol).

As soon as a node acquires a temporary color, it discards its permanent color, regardless of whether the latter is valid or not. The temporary color is used until it is replaced by a permanent one. Due to the contention nature of the FPRP protocol, a node may fail to obtain a temporary color. It can participate in the remaining part of this COF only if its permanent color is valid. Such a node contends for a temporary color, not because it needs a permanent color in the CFS, but because it needs new transmission slots in the IFS. If unsuccessful, it will have to suppress its requirements for new transmission slots until the next COF. If a node finds its permanent color corrupted due to some topological change, but fails to obtain a temporary color, it is not possible to proceed further in the current COF. It will have to wait for the next COF, and collisions could occur at this node during the interval in between [2]. Since this situation is undesirable, the algorithm's parameters, viz. the number of temporary colors and the number of reservation

protocol/mechanism monitoring the health of a link or flow.

[2]Collisions could also occur if two adjacent nodes obtain the same temporary color. This is possible, though rare, due to the non-perfect coloring characteristics of FPRP, especially when two adjacent nodes do not share a common neighbor.

cycles associated with them, should be chosen such that this occurs with a suitably low probability. Would it occur, it can be detected and handled by upper layer protocols as necessary, and can be viewed as a transient case of network unreliability.

The CFS is temporarily updated after the nodes acquire temporary colors. It has been brought back to a working condition, with the corruptions caused by topological changes resolved. Only with the CFS updated can the nodes communicate with each other freely. Suppose that a reasonable set of parameters are chosen and that every contender acquires a temporary color. The temporary colors and the permanent colors are different and do not conflict. As a result, by the end of the CR phase, every node has a color: either a temporary one it just acquired, or a permanent one assigned to it previously. All these colors represent different broadcast slots and will be used in the subsequent phase.

B.2 Allocation/Elimination Phase

In the Allocation/Elimination (AE) phase, nodes use the recently updated CFS to update the IFS. From the CR phase, every node has a conflict-free color (indeed a broadcast slot), and these broadcast slots are carefully interleaved so that nodes with temporary colors can choose their desired transmission slots and new permanent colors in a conflict-free fashion in the AE phase. During the AE phase, information slots are released from ending flows and assigned to newly arrived flows, or redistributed among on-going flows to resolve conflicts caused by nodal mobility. At the same time, the CFS itself is being refined. As new transmission slots are being *allocated*, the temporary colors are gradually being *eliminated* and replaced by permanent colors. This is why the second phase is called "Allocation/Elimination" phase.

An AE phase has t AE frames, AE1,...,AEt, each corresponding to a temporary color. Each AE frame is intended for nodes with a certain temporary color to choose their transmission slots and new permanent colors. The sizes of the AE frames vary slightly, with each successive frame being one slot shorter than its predecessor. We will describe the first such frame, AE1, in detail.

Assume that when frame AE1 begins, every node knows its own schedule, and the IDs and the colors (temporary or permanent) of all its one-hop neighbors. Frame AE1 starts with p slots, each of which is dedicated to nodes with a given permanent color. Nodes with permanent colors broadcast in turn in their designated slots, r, g, b, etc. In the slot designated to its color, a node broadcasts its IFS, including its schedule assignment (T, R, Br, Bt, Btr, or I), in each IS. It also broadcasts the IDs and the colors of itself and all its one-hop neighbors. Since nodes with permanent colors do not have requests for new transmission slots or new permanent colors, the schedules they broadcast are not much different than their previous schedules[3]. The only possible change is that a flow has ended and the transmitter node needs to release the slot, or that a node suffers a collision (multiple packets arrival) and the slot becomes $Br(X)$. To release a slot which it previously reserved, a node will not announce that slot as transmitting (T), but as idle (I) or blocked-from-receiving (Br), depending on whether it has a one-hop neighbor transmitting in the same slot (in this case Br) or not (in this case I). When a node broadcasts, all its one-hop neighbors listen and learn its most up-to-date schedule. From this information they can make changes in their own schedules accordingly. After the *p* slots for permanent colors, nodes with temporary colors broadcast their schedules in *reverse order*, from t to 1 (the rationale for this reversal will be explained shortly). When each node with temporary color t broadcasts, it transmits its schedules as did the nodes with permanent colors. Although such a node requires new transmission slots and a new permanent color (which is the reason it acquires a temporary color), it does not make any claim for such at this moment. After temporary color t is finished, each node with temporary color $t-1$ broadcasts and so on, until all nodes with temporary color 2 and higher have broadcasted, leaving only those nodes with temporary color 1 remaining. So far every node with temporary color 1 has received broadcasts from all its one-hop neighbors, and has learned the schedules of these nodes. It also knows the IDs of every node up to two hops away. Of course, it knows its own schedule. It is now able to choose its new transmission slots and a new permanent color. If it needs to reserve a new slot to transmit to a one-hop neighbor (or a set of one-hop neighbors in the case of multicast or broadcast), it looks for a slot in which the intended receiver(s) is RTR and itself is RTT. This slot will not conflict with any other transmissions. This node also picks a permanent color. Since it knows the color of every other node within two hops, the permanent color it chooses will not conflict with them. Note that nodes with the same temporary color may (and likely will) choose different permanent colors, depending on their neighborhoods. When choosing transmission slots and permanent color, a node uses a greedy algorithm. It chooses the *first* (the lowest indexed) available transmission slots and the *first* available permanent color. This results in efficient bandwidth utilization. In fact, any other algorithms which require only

[3]It should be possible to design an efficient packet format for transmitting only the *changes* to the schedules (not entire schedules every time), thus saving bandwidth.

local information can also be used.

At this point, nodes with temporary color 1 have fulfilled their requirements for new transmission slots and new permanent colors (or at least have had a chance to do so)[4]. The reason that the temporary colors are announced in reverse order, from p to 1, is to allow nodes with temporary color 1 to choose transmission slots and permanent colors *first*. This is because, among the sets of nodes with temporary colors, the set with temporary color 1—the *first* temporary color for which nodes contended—will most likely contain the *greatest number* of nodes. For this reason, this set of nodes is given the *highest* priority in slot and permanent color assignment. After these nodes have made their selections, all changes are incorporated into their latest schedules and broadcasted in the slot designated to temporary color 1. From now on these nodes discard temporary color 1 and start to use their new permanent colors. Temporary color 1 does not appear in this COF again, and we say it has been "eliminated".

AE2 is essentially identical to AE1, except that there is no slot dedicated to temporary color 1. Nodes again incorporate the latest updates into their schedules and broadcast to their neighbors. At the end of AE2, nodes with temporary color 2 choose new transmission slots and permanent colors, and broadcast their updated schedules to their neighbors. Temporary color 2 is eliminated afterwards. All the other AE frames follow in a similar manner. The nodes that are about to choose new transmission slots are always the *last* ones to broadcast in an AE frame. By the time they choose, they will always have received the broadcasts from all their one-hop neighbors, and have learned the latest schedules in their neighborhoods. They have enough information to select their transmission slots and permanent colors conflict-free. After each AE frame, nodes with a temporary color fulfill their requirements, and the corresponding temporary color disappears. As a consequence, every AE frame is a slot shorter than its predecessor. By the end of the last AE frame, the entire IFS has been updated. Flows which have ended have had their slots released, and flows which arose recently have been given transmission slots if enough bandwidth is available. The conflicts which are caused by node mobility have been resolved. The broadcast schedule of the network (CFS) has also been brought up to date. There are no temporary colors left. Every node has a permanent color which it can use in future COFs. Both the schedules (IFS and CFS) are conflict-free. The IFS is used in the following information frames until it is updated again in a subsequent COF.

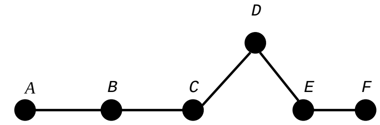
[4]It is more important to get a permanent color. When the network is heavily loaded, flows that are cannot be assigned transmission slots will have to be backlogged and attempt to acquire IS in subsequent COFs.

B.3 Network Initialization

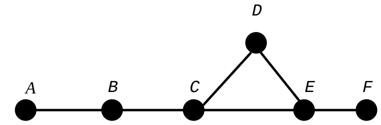
In the aforementioned procedure, we assume that the network has already been operating with some schedules (CFS and IFS), and that the E-TDMA protocol is used to update these schedules. The same procedure applies to a network which is just powered on. In this scenario, the only difference is that the old schedules are non-existent, and every scheduled transmission is new. Now, if all the nodes are turned on at *precisely* the same time, they will all contend for the temporary colors and many of them will fail at the beginning—these nodes cannot participate in the current COF. However, in reality, it would rarely be the case that every node is turned on precisely at the same time. When a node is powered on, unless it is the first node or an isolated node, it is likely to find other nearby nodes already running some schedules. By listening to their transmissions, this node get to learn the schedules in its neighborhood. It can then join the other nodes in the next COF. Even if all nodes were turned on simultaneously, this is not a deadlock problem for the evolutionary protocol. During every COF, some nodes will manage to obtain temporary colors and become involved in the schedule. Eventually, all the nodes will join the network and participate in the transmissions. A larger number of contenders merely increases the initialization delay.

IV. An Example

We now illustrate via example how the E-TDMA protocol can be used to update transmission schedules (Figure 2). There are 6 nodes (A to F) in the network, and the E-TDMA protocol has 2 temporary colors (1 and 2) and 4 permanent colors (r, g, b, y). There are 4 slots in an information frame (IS1 to IS4). The original topology is shown in Figure 2.a. Suppose the CFS and the IFS were conflict-free when they were generated according to the original topology, and these schedules are shown in Figure 2.c. Suppose node E moved towards node C and a new link appeared between them (Figure 2.b). This causes conflict in the original schedules, and the corrupted schedules are shown in Figure 2.d. Two on-going transmissions, from D to C in IS1 and from F to E in IS2, are corrupted, and they need to reserve new transmission slots for the on-going flows. We also assume that at the same time, node A needs to reserve a new transmission slot to transmit to node B. So we will see how the protocol reallocates conflicting transmissions and accommodate new ones. When the COF begins, the three nodes A, D and F, which require new transmission slots, contend actively for the temporary colors with the FPRP protocol. Assume they all succeed, and nodes A and D acquire temporary color 1 and node F



(a). Original topology



(b). Changed topology

	A	*B*	*C*	*D*	*E*	*F*
IS1	*T(B)*	*R(A)*	*R(D)*	*T(C)*	*T(F)*	*R(E)*
IS2	*Bt(B)*	*R(C)*	*T(B)*	*Btr(E;C)*	*R(F)*	*T(E)*
IS3	*R(B)*	*T(A)*	*Br(B)*	*I*	*I*	*I*
IS4	*I*	*I*	*I*	*I*	*I*	*I*
color	*r*	*g*	*y*	*b*	*r*	*g*

(c). Original Schedule

	A	*B*	*C*	*D*	*E*	*F*
IS1	*T(B)*	*R(A)*	*Btr(B;X)*	*T(C)*	*T(F)*	*R(E)*
IS2	*Bt(B)*	*R(C)*	*T(B)*	*Btr(E;C)*	*Br(X)*	*T(E)*
IS3	*R(B)*	*T(A)*	*Br(B)*	*I*	*I*	*I*
IS4	*I*	*I*	*I*	*I*	*I*	*I*
color	*1*	*g*	*y*	*1*	*r*	*2*

(d). Corrupted Schedule

	A	*B*	*C*	*D*	*E*	*F*
IS1	*T(B)*	*R(A)*	*Btr(B;E)*	*Br(E)*	*T(F)*	*R(E)*
IS2	*Bt(B)*	*R(C)*	*T(B)*	*Br(C)*	*Br(C)*	*I*
IS3	*R(B)*	*T(A)*	*Br(B)*	*I*	*I*	*I*
IS4	*T(B)*	*R(A)*	*R(D)*	*T(C)*	*Btr(C;D)*	*I*
color	*r*	*g*	*y*	*b*	*r*	*2*

(e). Partially updated schedule after AE1

	A	*B*	*C*	*D*	*E*	*F*
IS1	*T(B)*	*R(A)*	*Btr(B;E)*	*Br(E)*	*T(F)*	*R(E)*
IS2	*Bt(B)*	*R(C)*	*T(B)*	*Br(C)*	*Br(C)*	*I*
IS3	*R(B)*	*T(A)*	*Br(B)*	*Bt(E)*	*R(F)*	*T(E)*
IS4	*T(B)*	*R(A)*	*R(D)*	*T(C)*	*Btr(C;D)*	*I*
color	*r*	*g*	*y*	*b*	*r*	*g*

(f). Updated schedule after AE2

Fig. 2. The E-TDMA protocol in a simple network. There are four permanent colors (r, g, b, y) and two temporary colors (1,2).

acquires temporary color 2. In AE1, nodes A and D update their schedules after hearing broadcast from all their neighbors. Both of them schedule their transmissions in IS4. They also choose permanent colors and discard temporary color 1. Node A picks r and node D picks b. All these selections (transmission slots and permanent colors) are the first available ones (recall the greedy choice) given the network scenario. The partially updated schedules after AE1 are shown in Figure 2.e. In AE2, the node with temporary color 2 (node F) updates its schedule. It picks IS3 to transmit to node E as well as permanent color g. After AE2, both schedules (the CFS and IFS) are updated (Figure 2.f), with the conflicting transmissions reallocated to new slots and newly arrived transmission given its slot. Although only unicast transmissions are shown in the example, it is clear that multicasts and broadcasts can be handled in the same way.

V. Simulation Results

The performance of the E-TDMA protocol is studied with simulations. To start with, the protocol is examined in a fixed network with regular topology. 100 nodes are placed on a plane as a 10 by 10 square grid. A node is connected to its 4 neighbors in 4 different directions (north, south, east and west). A wireless channel of 1 Mbs is assumed. Time is slotted. The duration of an information slot is 1 ms. An information frame has $n = 20$ information slots, resulting in a frame length of 20 ms. The E-TDMA protocol is executed every $m = 10$ frames, i.e. there are 10 IFs between consecutive COFs. The number of permanent colors $p = 6$ and the number of temporary colors $t = 3$. Hence, an AE phase has $\sum_{i=1}^{t}(i + p) = 24$ slots. Four FPRP-based contention cycles are used to assign each temporary color. The length of a COF is roughly the same as an IF and hence, with $m = 10$, we estimate the overhead of the E-TDMA protocol as roughly 10% of the total bandwidth. This overhead can be reduced by increasing m, but the algorithm's reactivity would also be reduced as there would be a longer interval between schedule updates. User traffic, or flows, consist of streams of fixed-length, 125 byte packets, each fitting into an information slot. The number of packets per flow (i.e. the flow duration) is modeled as a geometric random variable with mean of 200 packets/flow. The generation of the flows at each node per slot is modeled as a Bernoulli random variable with probability p_B. Consequently, the number of arrivals at each node between 2 COFs is a Binomial random variable, which approximates a Poisson random variable with mean $A = p_B * m * n$. User-generated broadcast and unicast data traffic is modeled, where a broadcast is addressed to all the neighbors and a unicast is addressed to a randomly chosen neighbor. All the traffic are one hop transmissions, i.e., packets terminate at the receivers and are not forwarded further. Another way to look at it is that the end-to-end traffic is "de-hopped" at the MAC layer. The time-space correlations among these traffics are ignored, and the transmissions from different nodes are treated as i.i.d. random processes. When a flow arrives at a node, it is suspended until the next COF, after which it generates packets at a constant rate of 1 packet/frame. A packet has

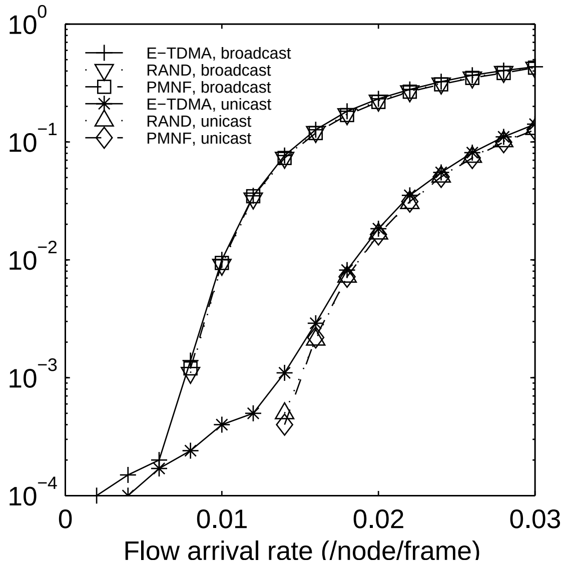


Fig. 3. Packet loss probabilities under single type traffic. The traffic consists of broadcast flows only (top) and unicast flows only (bottom).

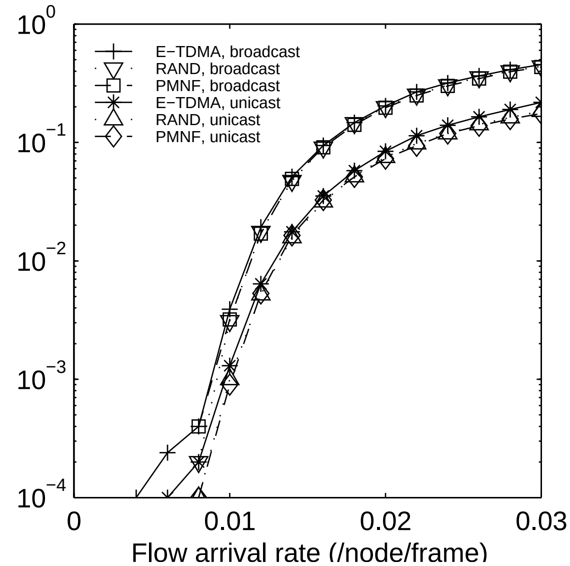


Fig. 4. Packets loss probabilities under mixed traffic. The traffic consists of broadcast and unicast flows at a 0.5:0.5 ratio.

a lifetime of 1 frame, and is dropped if it cannot be transmitted in a frame. Dropped packets are not retransmitted. When a node makes a reservation for its traffic, broadcasts are assigned transmission slots before unicasts, since they are destined to more receivers and are considered of higher priority. A flow is backlogged if it cannot be assigned a transmission slot during the current COF and, in the next COF, it is given priority over newly-arrived flows. It is possible, though rare, for collisions to occur in the schedules generated by the E-TDMA protocol. These collisions are caused by conflicts in temporary colors produced by the FPRP protocol. Two adjacent nodes may accidentally reserve the same temporary color, especially when they do not share a common neighbor. These nodes cannot coordinate with effectively in the E-TDMA protocol, since they always transmit in the same slot and are not aware of each other's schedule. Collisions may occur if they choose to transmit in the same slots. A unicast packet is considered lost if it collides at its receiver, and a broadcast packet is lost if it collides at any of its receivers. Transmissions suffering collision are not rescheduled and end naturally (the performance would improve if a collision is detected and the transmission is rescheduled). So two reasons contribute to packet loss here: packets dropped due to bandwidth unavailability and packet collisions due to imperfect scheduling. The packet loss probability is used as the performance measure.

The E-TDMA protocol is compared with two centralized TDMA scheduling algorithms, the "Progressive Minimum Neighbor First" algorithm (PMNF) and "Random Ordering" (RAND) algorithm [14]. The RAND algorithm generates a random ordering for the nodes, and processes their bandwidth requirements according to this random order. In the PMNF algorithm, nodes are assigned transmission slots in a systematic order, depending on the number of neighbors of each node and their connectivities. This algorithm is known to produce superior results for both broadcast and unicast scheduling. A greedy algorithm is used in both PMNF and RAND when a node chooses its transmission slots. The PMNF and the RAND algorithms are executed as frequently as the E-TDMA algorithm (once every 10 frames). The schedules produced with the centralized algorithms are always conflict-free, and the only reason a packet is lost is due to bandwidth unavailability. All the simulations are performed for 100,000 frames. Figure 3 shows the results for single type traffic and Figure 4 shows the results for mixed traffic.

The performances of the two centralized algorithms are very close, with the PMNF performing slightly better. Under the same load, broadcast traffic has a much higher packet loss rate than unicast traffic, even though it is given a higher priority. This is because broadcast utilizes the spectrum more heavily. To schedule a broadcast transmission, all the receivers and the transmitter must be available in the same slot, a requirement more demanding than a unicast. As the traffic gets heavy, much of the bandwidth becomes fragmented and unusable for broadcast transmission. This is due, in large part, to unicast transmissions which inhibit many broadcasts. A unicast transmission is easier to schedule as it has fewer receiver constraints, and can grab a slot unschedulable for a broadcast, thereby fragmenting the bandwidth and blocking subsequent broadcasts at neighboring nodes.

The E-TDMA protocol performs similarly to the centralized algorithms. Only in the very lightly-loaded regime does the E-TDMA protocol suffer any noticeable degradation relative to the centralized approaches. This is because under very light traffic, almost all transmissions are immediately assigned bandwidth by the centralized algorithms, thus few packets are dropped. In the E-TDMA protocol, a node may fail to obtain a temporary color with contention, causing packets to be dropped until the next COF, or obtain a temporary color conflicting with one of its neighbors, causing collisions in the schedule and all the transmitted packets lost. In both cases, the source of error is the contention-based, non-perfect coloring characteristics of the FPRP protocol. These packet losses would be eliminated if the FPRP were replaced by a contention-free, perfect coloring protocol with the same distributed, scalable properties. When we simulated E-TDMA with a centralized, random coloring protocol in the place of FPRP, the the performance became very close to the centralized algorithms (these results are not shown due to space constraints). In fact, the square grid topology we have used is the worst case of those topologies we simulated for E-TDMA because two adjacent nodes do not share a common neighbor. Such neighbor sharing is important to reduce collisions in the temporary coloring produced by FPRP protocol. As E-TDMA here depends on the FPRP, its performance is hurt by this shortcoming of FPRP. In a densely connected network, collisions become much less likely, as we have seen from simulations in a network with a hexagonal topology. In this case, packet collisions due to imperfect scheduling are negligible. Even in the square topology, since the packet loss probability is very low (less than 0.1%) in the region where E-TDMA protocol degrades relatively to PMNF and RAND, we conclude that the packet loss due to imperfect scheduling is insignificant when compared with other sources of transmission error in these networks, and does not pose a problem for the applicability of the E-TDMA protocol operating with the FPRP. As the load becomes heavy, all three protocols perform essentially the same, since the main source of packet loss is the limited bandwidth. This implies that the qualities of the schedules generated by these protocols are very close. This is not surprising. As a matter of fact, the performance of the E-TDMA algorithm converges to that of the RAND algorithm as the number of permanent and temporary colors increase, because the FPRP algorithm (used in the CR phase) is a random coloring (ordering) algorithm, and the operation in the AE phase employs a deterministic, greedy algorithm to assign the slots. If there are enough temporary colors in a COF, every node requiring new transmission slots can obtain a temporary color, the order of which is random, and the slots are then assigned in a greedy fashion. It is not hard to see that this is equivalent to the RAND algorithm, just implemented in a distributed fashion.

Node mobility is not included in the simulations presented here. With an interval of $200ms$ between two COFs, we judge the schedules are updated frequently enough to cope with network dynamics under moderate mobility. Even when the network becomes too volatile to be handled by the protocol, schedules can be generated quickly once the network becomes relatively stable. The E-TDMA protocol starts to degrade slightly at heavy load. This is because when new flows arrive more frequently, more nodes require temporary colors in each COF, and the limited number of temporary colors (3 in the simulations) cannot accommodate all these simultaneous requests. Some of them have to be backlogged and packets are dropped meanwhile. This is a penalty paid for fixing the protocol parameters, and can be resolved if the parameters are adjusted dynamically to suit the traffic and the network topology—such adjustment is the subject of current work. We also conducted simulations in a larger network (with 400 nodes). As with the smaller network, the performance of the E-TDMA protocol is very close to that of the centralized algorithms. While the overhead of the centralized algorithms grows with the network size, that of the E-TDMA protocol stays constant. More simulations are under way, including the effects of different traffic patterns, network dynamics, node mobility and network heterogeneity.

VI. Discussions

The E-TDMA protocol is unique in that it maintains two transmission schedules at the same time. These schedules are best suited to carry constant-bit-rate, real-time traffic, such as voice or video packets. The protocol provides a basis upon which QoS support for ad hoc networks can be built. The E-TDMA protocol provides partial information to support hop-by-hop QoS routing, because a node knows the schedules in its neighborhood and therefore the transmission state of every link. A query-based, on-demand routing algorithm is suitable here, since route-discovery and resource-reservation are jointly carried out hop-by-hop, preventing the situation when all the links along a route try to reserve bandwidth the same time. The latter is undesirable because it requires adjacent nodes to contend simultaneously, causing too much contention in the same CR phase. The time to setup a route is roughly the time to reserve the bandwidth along all the hops in the route. Although here we assume a single channel TDMA system, the E-TDMA protocol easily extends to multi-channel systems, where a channel can be dedicated solely for broadcast network control traffic, including the E-TDMA protocol, while other channels are used for user traffic. Priority of different traffic types can be supported in the protocol. Among nodes which obtain temporary colors to acquire new transmission bandwidth, temporary color 1 represents the highest priority and the others follow in descending order. A node with a high-priority color chooses its transmission slots *before* a node with a low-priority color. However, a node acquires its temporary color by contention, and this color is non-deterministic. Priority can be introduced probabilistically in the CR phase if a node contends for a high-priority color more vigorously when it has a higher priority. When a node contends, it needs to obtain only *one* color regardless of how many information slots it requires. This differentiates it from other contention-based schemes, such as the Packet Reservation Multiple Access protocol [18], where a node contends multiple times if it needs multiple slots. Although the protocol will work for arbitrary topology, it works best for a relatively homogeneous network. Here homogeneity means that the nodal degree is roughly uniform across the entire network. This is because the optimal choice of the parameters, especially the number of permanent colors, depends heavily on nodal degree. If the network remains relatively homogeneous as it grows in size, the performance of the protocol would be essentially unaffected. In the current protocol, a situation where too many nodes come close together—perhaps forming a "local clique"—will be handled poorly, since there probably would not be enough permanent colors to color all of them. We plan to design a mechanism which allows the parameters of the protocol to adjust to the network autonomously. A scheme which adjusts the transmission power (range) of the nodes depending on local nodal density (thus adjusting the local connectivity) would also help here in maintaining node degree uniformity.

VII. Conclusion

We have developed a new TDMA scheduling protocol for ad hoc networks. It is a fully-distributed and parallel algorithm, where the schedules are simultaneously generated and maintained over the entire network. Two schedules are kept by the protocol, one for network control and the other for user traffic. These schedules are frequently updated in an evolutionary manner to accommodate changes in network topology and bandwidth requirements. The operation of the schedule is not affected by the network size, and it is a scalable protocol suitable for networks of large size. Its performance has been studied with simulation. The results showed that the E-TDMA protocol works very well. Its performance is comparably to that of centralized algorithms.

References

[1] Institute of Electrical and Electronics Engineers, 445 Hoes Lane, P.O. Box 1331, Piscataway, NJ, 08855, USA, *"IEEE P802.11 Draft Standard for Wireless LAN: Medium Access Control (MAC) and Physical Layer (PHY) Specification"*, draft p802.11d5.0 edition, July 1996.

[2] European Telecommunications Standards Institute, 06921 Sophia Antipolis Cedex - France, *"Radio Equipment and Systems (RES): HIgh PErformance Radio Local Area Network (HIPERLAN), Type 1 Functional Specification"*, draft pr ets 300 652 edition, December 1995.

[3] I. Chlamtac and S. Kutten, "A spacial reuse tdma/fdma for mobile multi-hop radio network," Proc. IEEE INFOCOM, 1985.

[4] M. Post, A. Kershenbaum, and P. Sarachik, ""A Distributed Evolutionary Algorithm for Reorganizing Network Communications"," in *Proc. MILCOM*, 1985, pp. 133–139.

[5] A. Ephremides, J. Wieselthier, and D. Baker, "A design concept for reliable mobile radio networks with frequency hopping signaling," *Proceedings of the IEEE*, vol. 75, no. 1, January 1987.

[6] I. Chlamtac and S. Pinter, "Distributed nodes organization algorithm for channel access in a multihop dynamic radio network," *IEEE Trans. on Computers*, vol. C-36, pp. 728–737, 1987.

[7] A. Ephremides and T. Truong, "Scheduling broadcasts in multihop radio networks," *IEEE Transactions on Communications*, vol. COM-38, no. 4, pp. 456–460, April 1990.

[8] I. Cidon and M. Sidi, "Distributed assignment algorithms for multihop packet radio networks," *IEEE Trans. on Computers*, vol. 38, pp. 1353–1361, 1989.

[9] L. C. Pond and V. O. K. Li, "A distributed time-slot assignment protocol for mobile multi-hop broadcast packet radio networks," in *Proc. IEEE MILCOM*, 1989, pp. 70–74.

[10] A. Bhatnagar and T. Robertazzi, ""Layer Net: A New Self-Organizing Network Protocol"," in *Proc. MILCOM*, 1990, pp. 845–850.

[11] R. Ramaswami and K. K. Parhi, "Distributed scheduling of broadcast in a radio network," in *Proc. IEEE INFOCOM*, 1989, pp. 497–504.

[12] S. Ramanathan and E. L. Lloyd, "Scheduling algorithms for multihop radio networks," *IEEE/ACM Trans. on Networking*, vol. 1, pp. 166–177, 1993.

[13] I. Chlamtac and A. Farago, "Making transmission schedules immune to topology changes in multi-hop packet radio networks," *IEEE/ACM Trans. on Networking*, vol. 2, pp. 23–29, 1994.

[14] S. Ramanathan, ""A Unified Framework and Algorithm for (T/F/C)DMA Channel Assignment in Wireless Networks"," in *Proc. INFOCOM*, Kobe, Japan, April 1997.

[15] C. R. Lin and M. Gerla, "Adpative clustering for mobile wireless networks," *IEEE J. Select. Areas Commun*, vol. 15, pp. 1265–1275, 1997.

[16] C. Zhu and M.S. Corson, "A five phase reservation protocol (FPRP) for mobile ad hoc networks," in *Proc. IEEE INFOCOM*, 1998.

[17] E. Arikan, "Some complexity results about packet radio networks," *IEEE Trans. on Inform. Theory*, vol. IT-30, pp. 681–685, 84.

[18] D. J. Goodman, R. A. Valenzuela, K. T. Gayliard, and B. Ramamurthi, "Packet reservation multiple access for local wireless communications," *IEEE Transactions on Communications*, vol. 37, no. 8, pp. 885–890, August 1989.