Distributed Formation of Broadcast Transmission Schedules for Mobile Ad Hoc Networks
Brian J. Wolf, Joseph L. Hammond, Daniel L. Noneaker, and Harlan B. Russell
Dept. of Electrical and Computer Engineering, Clemson University, Clemson SC

Abstract: In a mobile ad hoc packet radio network using transmission scheduling, when a terminal boots and is ready to join the network it has no knowledge of the network’s current state. In order to send and receive packets without interference, the new terminal must have some means of notifying other terminals of its presence and exchanging information with them so that it can form a collision free broadcast transmission schedule. We demonstrate the performance of a distributed protocol designed to allow a new terminal to acquire sufficient information about its local environment to achieve collision free transmissions. Furthermore, we show that changes to the network are limited to the new terminal’s local neighborhood. As a result, the size of the overall network has very little effect on the performance of the protocol. The protocol is not only effective in adding a new terminal; it can also organize a group of terminals into a new network. This is demonstrated for the extreme case of all terminals powering up simultaneously. The overall performance of the algorithm depends primarily on the density of the network and rate at which changes occur in the network.

1. Introduction

In a mobile, multi-hop packet radio network frequent connectivity changes place high demands on channel access protocols. Transmissions are broadcast using omnidirectional antennas, so media access control (MAC) protocols are required to coordinate transmissions among the terminals to avoid excessive levels of multiple-access interference. One class of MAC protocols often used in ad hoc networks is contention-based protocols. However, it is well known that such protocols can often lead to poor performance under heavy traffic loads and are unable to guarantee regular access to the channel. Another class of protocols, called scheduling protocols [7], operate by establishing transmission schedules in which each terminal is assigned transmission times so as to avoid contention for the channel or the intended receivers. Transmission scheduling protocols make it possible to guarantee a certain quality of service because terminals can reserve regularly recurring transmission slots.

Scheduling protocols that can provide contention-free link-level broadcast transmissions in a mobile ad hoc network are available, however, many implementations are suitable only for relatively static network topologies, are not distributed, or cannot adapt quickly to changes in topology (e.g., [5], [6], [12]). Several recent papers have presented distributed broadcast-scheduling protocols that can adapt locally to changes in topology (e.g., [3], [10], [11], [13], and [14]). In all of these protocols, a terminal can decide whether or not to transmit based solely on its knowledge of the transmissions of terminals one or two hops away. Since scheduling decisions can be made on the basis of this local information, there is no need for a single terminal to coordinate the transmissions of the network as a whole; control can be distributed to the individual terminals. In [1] we presented and evaluated our version of a distributed, adaptive transmission scheduling protocol. Our approach differs from others in the use of a frame size that depends on a terminal’s local neighborhood only and an efficient algorithm to assign transmission slots. In addition, the protocol does not need a contention-based random-access phase to respond to changes in connectivity.

In this paper we develop and evaluate an initialization protocol that forms the initial slot assignments and incorporates new terminals into the network. When terminals move, leave the network (e.g., power down, enter a sleep mode, or affiliate with a different network), or join the network (e.g., boot up or otherwise wish to begin participating in the networking protocols), the transmission schedule must be adapted to maintain content free transmissions. The transmission scheduling protocols should be able to recognize changes and adapt the schedule accordingly to maintain high channel utilization. Our previous investigations [1]–[2] have focused on the scenario in which the network is already initialized and the nodes that are participating in the network have already established a transmission schedule. The adaptive protocols are designed to adapt quickly to changes in the network topology. These protocols are not designed to integrate into the network a terminal that does not already have some initial transmission assignment.

In practice, the terminals have to power up at some point, and in some situations, the terminals may even power up and power down with some regularity. Powering down is easily handled by requiring the terminal to broadcast a special message to its neighbors before it turns off. Terminals in the vicinity simply reclaim the slots formerly used by the vacating terminal, and this is handled by the delete-connection protocol described in [1]. From the point of view of the network, a terminal powering up is analogous to several communications links being added simultaneously. However, from the point of view of the new terminal the situation is far more complicated. The

This work was supported by the Multidisciplinary University Research Initiative (MURI) program administered by the Office of Naval Research under Grant N00014-00-1-0565 and by NSF under Grant ANI-0092986
new terminal will initially have no knowledge of the network timing or its local neighborhood. Assuming the terminal can monitor the channel to detect the timing of transmissions, we are still faced with a daunting task since forming a new link requires not only the terminals on both ends of the link to adjust their transmission assignments but also all terminals in the local area as well. Consequently, if the new terminal has $n$ neighbors, the terminal must establish communications with each neighbor. If $n$ is larger than two or three, the overhead required to adjust the transmission schedules with the adaptive protocols is unacceptable.

To solve these problems, we have developed the transmission schedule construction protocol (TSCP). The remainder of the paper is structured as follows. Section 2 describes the network model and the distributed and adaptive transmission scheduling protocol that adapts transmission assignments to changes in connectivity. The details of the TSCP are described in Section 3. In Section 4 the protocol performance is evaluated in the case of a single node powering up in a static network, considering variables such as network size and density. We then evaluate the performance of the algorithm for an extreme network initialization scenario in which all network terminals power up simultaneously. Finally, Section 5 gives conclusions.

2. Network Model

We consider a network of terminals that employ half-duplex radio communication and identical transmission capability for all terminals, and we assume that each link is symmetric. Time is divided into slots and all terminals are synchronized with the time-slot boundaries. In the absence of transmission interference, two terminals that are neighbors can communicate reliably. If more than one neighbor of a terminal transmits simultaneously, then the terminal cannot receive any of the transmissions. A terminal in the receive mode cannot distinguish between an idle channel and a collision.

The adaptive slot assignment protocol uses a distributed algorithm due to Lyui [9] for assigning transmission slots. A local neighborhood of a terminal $k$ is defined as the set, $N(k)$, of all terminals in direct communication with terminal $k$, termed 1-neighbors of terminal $k$, and the 1-neighbors of terminal $k$’s 1-neighbors, termed 2-neighbors of terminal $k$. Using the distributed algorithm, a terminal $k$ selects a color number, $c_k$, which is the smallest positive integer that is different from the color numbers of all of the terminals in its local neighborhood. Terminal $k$ determines if it can transmit in slot $t$ by calculating the set of candidate terminals in its local neighborhood, $C(t,k) \subseteq N(k)$, that are eligible to transmit in this slot. Terminal $m$ is a candidate and contained in $C(t,k)$ if $t \mod p(c_m) = c_m \mod p(c_m)$, where $p(c_m)$ is equal to the smallest power of two greater than or equal to $c_m$. Finally, terminal $k$ transmits in slot $t$ if $k$ is a candidate for this slot and if it has the largest color number among the candidates (i.e., if $k \in C(t,k)$ and $c_k > c_m$ for all other $m \in C(t,k)$). Each terminal has its own frame size that is the smallest power of two greater than or equal to the maximum color number in its local neighborhood. Notice that for a terminal to determine its frame size and in which slots to transmit, it only needs to know the terminals in its local neighborhood and their color numbers.

It is shown in [9] that the assigned slots result in collision-free transmissions and each terminal is guaranteed at least one assigned slot in each of its frames. In addition to its guaranteed slot, a terminal may be assigned additional slots within its frame depending on its color number and the color numbers of the other terminals in its neighborhood. A concise formulation of Lyui’s algorithm and a discussion of the properties of the transmission assignments are given in [8].

Each time slot is subdivided into an identification interval and a payload interval, as illustrated in Figure 1. The identification interval is used to transmit a short packet containing flag bits to identify the transmitter and its color number and frame size. The identification interval is part of the mechanism used to identify collisions due to changes in connectivity or incorrect coloring information.

![Frame format for terminal m](image)

**Figure 1.** Frame format for terminal $m$, where $c_p$ is the largest color number in terminal $m$’s local neighborhood.

A terminal that is assigned to transmit in a time slot will transmit either a data or control packet during the payload interval. In addition, during the identification interval of this slot, the terminal either transmits a short packet containing the flag bits or it remains in the receive mode. For each slot in which a terminal transmits a data packet in the payload interval, first it transmits the flag bits with a probability of 0.5 in the identification interval. On the other hand, if a terminal transmits a control packet in the payload interval of a slot, it always first transmits the flag bits during the identification interval, unless the terminal has determined that there was a collision in a previous interval. In the latter situation, the control packet is used to notify the terminal’s neighbors of the collision, and the flag bits are first transmitted only with a probability of 0.5.
The TSCP, which is the subject of this paper, is tightly integrated with the add-connection and delete-connection protocols defined and evaluated in [1]. The two distributed algorithms for adding and deleting links employ Lyui’s basic slot assignment method to adjust slot assignments using only information exchanged in the local neighborhood. Simulation studies, using a large number of randomly generated networks with different diameters, degrees, and number of terminals show that the algorithms function properly and are efficient in the assignment of slots. The simulations also show that the algorithms adapt slot assignments to added and deleted links in reasonable convergence times, which average from 40 to 100 slots. Using variable density graphs, average convergence time is shown to increase in almost a linear fashion with terminal radius which, in turn, is closely related to average degree. Studies with constant density graphs show that convergence time is relatively insensitive to the number of terminals.

3. The Transmission Schedule Construction Protocol

The TSCP is designed to enable a single terminal to integrate itself into an existing network by constructing a collision-free transmission schedule for itself and indicating what changes, if any, are required at its neighbors. We describe the TSCP for a new terminal that is ready to determine its neighbors and initiate communication with them. The operation of the neighboring terminals of the new terminal is also described. A consequence of the protocol is that a collection of arbitrarily distributed terminals can self organize into a network. The actions taken can be divided into three phases, illustrated as a flow chart in Figure 2.

Phase 1. The terminal joining a network detects the terminals that will be its neighbors. The first step requires a terminal to remain in the receive mode to learn about as many neighbors as possible through overhearing packets and waiting for an opportunity to initiate the protocol. The next step involves an iterative process to try and contact as many of its neighbors as possible to establish a temporary polling schedule.

Phase 2. The new terminal establishes a temporary schedule to contact each of the neighbors it has identified to collect state information.

Phase 3. The new terminal selects a color number and creates a transmission schedule for itself and determines if any of its 1-neighbors need to change their color numbers. The color number for the new terminal and its 1-neighbors is forwarded to the new terminal’s 1-neighbors. Each 1-neighbor forwards a control packet to its 2-neighbors. A 2-neighbor does not need to change its color number, however it forwards a control packet to its neighbors if it has learned of a new color number from any of its 1-neighbors.

The completion of these tasks incorporates the new terminal as a part of a network allowing normal, contention-free transmissions to take place. We now elaborate on the actions required in each of the phases to incorporate the new terminal into a network.

A new terminal begins to join the network by remaining in the receive mode for several slots to synchronize to the timeslot boundaries. The terminal then monitors the identification interval for flag bits, attempting to determine the identification of those neighbors for which it can receive the flag bits without collisions. The new terminal also attempts to determine if one of its neighbors is in the process of executing the TSCP, or the add-connection or delete-connection protocols. If so, the terminal selects a random back-off delay. Otherwise, it begins a series of signal/acknowledgement (S/A) handshakes to identify as many of its 1-neighbors as it can. An S/A handshake consists of 4 consecutive time slots in which the new terminal may transmit up to 4 control packets, followed by one slot for neighbors hearing the control packet to reply with an acknowledgement. The new terminal transmits in 1 of the 4 slots in the first S/A sequence, 2 of the 4 slots in the second S/A sequence, 3 of the 4 slots in the next sequence, and all four slots in subsequent S/A sequences. A neighbor can detect the new terminal if it receives the entire control packet or if it just receives the flag bits.
transmission slice is 1/16 of a slot, although this choice can
experiments have shown that a reasonable value for a
considerable number fit into a standard slot. Simulation
transmission slice is a minipacket short enough so that a
with a small probability of collision, and only those minipackets
transmitted in a slice without a collision can be received by
the new terminal. After the new terminal processes the
The S/A handshakes continue until several sequences
have passed in which the new terminal does not detect any
new neighbors. During this time, 1-neighbors have at least
one opportunity to send a defer message to their 2-
eighbors in an assigned slot, and then defer themselves.
The probability that a neighbor that has not detected the
new terminal receives an S message increases with each
iteration of the S/A sequence as more 1 and 2-neighbors
defer from transmitting.

Phase 2 begins when the new terminal has determined it
has learned of as many of its 1-neighbors as possible (some
terminals may not be detected due to repeated collisions).
The new terminal develops a temporary polling schedule
for the local neighborhood assigning each neighbor a
time slot, and the new terminal broadcasts this
its 1-neighbors. Using these temporary slots, 1-
eighbors reply, in turn, with their own color numbers and
their neighbor’s color numbers. As the new terminal hears
from each 1-neighbor, it computes a list of possible colors
that the neighbor can take on without causing conflicts in
its own neighborhood.

Phase 3 begins after each neighbor of the new terminal
has been heard from. The new terminal uses its list of
possible colors for each neighboring terminal to resolve
any color conflicts. Finally, it chooses a color number and
transmission slot for itself and broadcasts all of the new
color assignments to its 1-neighbors. These 1-neighbors
use their color numbers to modify their transmission slots,
if necessary, and then use their assigned transmission slots
to broadcast the new schedule to their neighbors, (the new
terminal’s 2-neighbors). Finally, updates are completed by
transmitting the new information from the 2-neighbors of
the new terminal to terminals three hops away.

The description of the basic operation of the TSCP
given above is based on the assumption that all terminals
are free to interact. In a general environment, however,
terminals may be processing other protocols, to adapt to the
environment, to correct errors etc., and thus may not be in a
state to incorporate new terminals. Thus throughout each
phase of operation, as described above, the new terminal
must also monitor the state of the local neighborhood to
detect the execution of other protocols which may be
adding a new link or even executing the TSCP through
another terminal. Overlapping changes in close proximity
can lead to invalid data and interfering transmissions.
There are many points in the protocol at which this may
happen, and different responses the protocol can take.
Space does not permit a complete examination of all
possible cases. However, the most common response is for
the new terminal to suspend operation of the TSCP until
the other change or changes have been responded to.
Another possibility is for the new terminal to complete
execution of the TSCP, and force the other parts of the
network to respond to competing changes at a later time.

4. Evaluation of the Transmission Scheduling
Construction Protocol

A computer simulation is utilized to evaluate the
performance of the TSCP. First, we study the scenario in
which one terminal integrates itself to an existing network.
This is followed by examination of an initialization
scenario in which a number of terminals power up at the
same time and attempt to organize a network.

For the simulation studies, we assume that if a
transmitter is within a specified distance of a receiver, the
transmission attempt is successful if there is no multiple-
access interference and is unsuccessful otherwise.
Transmission errors due to factors other than collisions are
not included in the simulation results reported in this
manuscript. The network for a particular simulation trial
consists of a specified number of terminals within a square
area, and with fixed, identical, transmission radii for all
terminals. For each simulation trial, the terminals are
placed at randomly chosen locations with a uniform
distribution within the specified area. We consider two
different classes of networks: constant density and variable
density. For the constant density networks, the ratio of the
number of terminals to area is fixed at 1/100^2 (terminals per
m^2) and the transmission radius is 200 meters. As the
number of terminals is increased for each simulation trial
so is the area so that the ratio of terminals to area is fixed.
For the variable density networks, the area is fixed to 1410 by 1410 and the number of terminals is 200. For simulations with variable-density networks, the transmission radius is specified for each trial. For both classes of networks, 100 different randomly chosen topologies are simulated for each simulation trial.

**Adding One Terminal to an Existing Network**

We simulate a single terminal powering up in an existing static network to explore the basic functionality and characteristics of the protocol. One terminal is chosen at random for a simulation trial to be the new terminal. All the other terminals are assumed to already have consistent transmission schedules, and the simulation assigns initial color numbers and calculates the transmission schedules. We then simulate the TSCP as the new terminal integrates itself into the network. The process of a terminal synchronizing itself to the time slot boundaries is not simulated.

![Graph 1](image1.png)

**Figure 3.** Average convergence time and number of terminals involved in updating transmission schedules for the constant-density networks.

We define *convergence time* as the time from powering up until the new terminal and all other terminals have collision-free transmission assignments. Results, using the constant density networks, are shown in Figure 3, where convergence time is plotted versus the number of terminals. Also shown in the figure versus the same independent variable is the number of terminals participating in the protocol. Note that both curves increase rapidly with the number of terminals until the number of terminals is greater than approximately 50. This behavior can be explained as follows. With fewer terminals, the network is generated on a smaller grid, so that the new terminal must interact with almost every other terminal when it powers up. Thus, as the number of terminals increases, more time is required for the new terminal to coordinate access with its neighbors before it is incorporated into the network and, of course, more terminals are participating. As the number of terminals increases beyond 50, however, only a portion of the network needs to interact with the new terminal and thus both the convergence time and the number of participating terminals is not sensitive to the network size.

We also studied the effect of network density on the performance of the protocol using the variable density networks and allowing a randomly chosen terminal to power up in the same manner as for the constant density case. Figure 4 gives a plot of convergence time for this case versus the transmission radius. The almost uniform increase of convergence time in this case with the transmission radius is explained by the fact that the number of neighbors with which the new terminal must interact also increases in direct proportion to the transmission radius. The departure from an exact linear increase is due to the quantized changes in frame size that accompany the use of Lyui’s algorithm. As noted in Section 2, Lyui’s algorithm assigns slots from color numbers in frames whose length must be powers of 2. Color numbers increase almost linearly with network density but frame sizes are quantized to powers of 2. The frame size affects convergence time, since terminals executing the protocol must transmit in their assigned slots.

![Graph 2](image2.png)

**Figure 4.** Average convergence time and number of terminals involved in updating transmission schedules for the variable-density networks.

**Network Self Organization**

The TSCP is not only effective in adding a new terminal to an existing network, it can also be used to organize a group of terminals into a new network. The ability to do this is evaluated for the extreme case that all terminals involved power up simultaneously. This is considered to be an extreme case because in typical operation the terminals will power up at intervals over a reasonable period of time and will not do so at the same
time. Powering up at the same time forces all of the terminals to contend for the features of the protocol which incorporate them into the evolving network. In this environment there are numerous collisions and conflicting color numbers can be selected by the distributed protocol as terminals independently attempt to claim the same color number. While the TSCP attempts to assign collision-free transmission times, there will be errors in the assignments. These errors are resolved with the add-connection and delete-connection protocols because the effects of collisions due to inconsistent transmission assignments are the same as due to collisions caused by mobility.

We evaluate the TSCP in conjunction with the add-connection and delete-connection protocols described in Section 2. Our simulations are based on variable density networks with 200 terminals powering up simultaneously. Each simulation trial is repeated 100 times and the results are averaged. Two criteria are used for evaluation: the number of calls to the TSCP and to each of the add-connection and delete-connection protocols, and the percentage of terminals with final, contention-free schedules.

Figure 5 shows the number of slots required to form a collision-free transmission schedule. The curves show percent of terminals transmitting successfully as a function of timeslot number for transmission radii of 150 and 200. The results show that the transmission schedules are 90% complete after approximately 2000 slots for a transmission radius of 150 and after approximately 5000 slots for a transmission radius of 200. Figure 6 shows the number of calls to the three programs as a function of slot number. The number of calls is quantized into the value for 50 time slot intervals. Results are shown for terminal transmission radii of 150 and 200.

The graphs in Figure 6 track the number of calls to the TSCP, the add-connection, and the delete-connection protocols as the network evolves. Each data point represents the number of calls to a protocol in a period of 50 slots, averaged over all simulations. As more terminals add themselves to the network, the average number of calls per 50-slot period tends to decrease. The peak values tend to be higher for simulations at a radius of 150, representing greater spatial reuse. Since terminals contend during phase 1 of the protocol, some terminals will have to wait a considerable amount of time before initiating the protocol themselves. This behavior is reflected by the rather long period of minimal activity near the end of the simulations; it should be pointed out that most terminals in the network are already transmitting with contention free schedules by this time.

![Figure 5. Percentage of terminals with a final contention-free transmission schedule as network initialization progresses.](image)

![Figure 6. Number of calls to TSCP, delete-connection, and add-connection protocols.](image)
5. Conclusions

We have developed a protocol that can establish a new collision-free transmission schedule for a group of terminals, or make it possible for a terminal to join an existing network. No prior knowledge of network size or number of terminals is required. The protocol is distributed and a terminal must exchange information only within its local neighborhood, thus allowing spatial reuse of the transmission medium. These properties are inherent in the design of the TSCP Protocol and are verified with simulation studies.

Simulation studies of a single terminal joining an existing network of terminals with fixed transmission radius are used to show that the protocol scales well. Results of simulation show that the number of slots required for convergence to a new schedule increases to approximately 120 slots as network size increases from 5 to 75 terminals and then remains essentially constant as network size is further increased to 250 terminals. At a network size of 75 terminals approximately 30% of the terminals participate in the actions of the protocol, while at a network size of 250 terminals only approximately 11% participate. In other studies, convergence time is shown to increase from 50 slots to 350 slots in an almost linear fashion over a range of transmission radii from 100 to 400.

Simulation is used to determine the number of slots required for 200 randomly located terminals to form a transmission schedule in a worst case for which all terminals power up at the same time. Results show that 90% of terminals have a collision-free transmission schedule after approximately 2000 slots for a transmission radius of 150 or 5000 slots for a transmission radius of 200. Corresponding values for 100% of terminals to have such schedules are 3000 and 9000 slots respectively.

It is useful to introduce some parameter values to aid in interpreting the results. Depending on the particular combination of symbol rate and packet size, the duration of a time slot is typically between 0.1 and 1 msec. This implies that if 200 terminals attempt to self organize starting at the same moment, the initialization time is between 0.9 and 9 seconds.

6. References