Distributed Assignment Algorithms for Multihop Packet Radio Networks

ISRAEL CIDON, MEMBER, IEEE, AND MOSHE SIDI, SENIOR MEMBER, IEEE

Abstract—**New distributed dynamic channel assignment algorithms for a multihop packet radio network are introduced. The algorithms ensure conflict-free transmissions by the nodes of the network. The basic idea of the algorithms is to split the shared channel into a control segment and a transmission segment. The control segment is used to avoid conflicts among nodes and to increase the utilization of the transmission segment. We also show how these algorithms can be used in order to determine TDMA cycles with spatial reuse of the channel.**

Index Terms—**Distributed algorithms, multihop packet radio networks, slot assignment, spatial channel reuse, time division multiple access (TDMA).**

I. Introduction

PACKET radio was introduced more than a decade ago in order to apply packet switching communications to a shared radio channel. The objective of packet radio is to supply data communications services to a population of geographically distributed, possibly mobile, radio units. In order to provide this service, the network must include a channel access protocol whose role is to allow nodes to multiplex their traffic on the shared channel. In addition, the access protocol must include management functions that track all nodes and route data reliably in spite of topological changes. The main difficulties in the design of packet radio networks stem from the unique features of this type of networks, namely, the multihop shared channel where not all nodes are within direct radio connectivity of each other, and the dynamic network topology due to mobility of nodes.

In this paper, we focus on the channel access assignment in a multihop network. The problem that we are faced with is how to share and control access to the common channel in a fashion that provides an acceptable level of performance. Most of the known channel access protocols were developed and optimized for a single-hop channel [13]. Consequently, they are either inapplicable or have many deficiencies when applied to a multihop environment.

In this paper, we propose a new *distributed dynamic channel assignment algorithm* which is *conflict-free* and suitable for shared channel multihop networks. By avoiding conflicts, we ensure a locally high channel utilization. By a distributed algorithm we mean that all nodes in the network execute identical algorithms and each node's decisions whether to access the channel or not are based only upon locally available information.

The idea behind the proposed algorithms is to split the shared channel into a control segment and a transmission segment, similarly to [2]. The control segment is used to avoid conflicts among the nodes and to increase the utilization of the transmission segment.

The issue of channel assignment has been the subject of several recent studies. Hajek and Sasaki [3] developed a polynomial-time algorithm that computes a minimum length TDMA cycle for a CDMA model where secondary conflicts are permitted. The complexity of their algorithm has been improved in [4] but it remains high in spite of it being polynomial. For the model where secondary conflicts are not permitted, a solution based on determination of the maximum cliques in the network is proposed in [5]. Simple heuristic algorithms for models with and without secondary conflicts have been proposed in [6]. Simulations have shown that these algorithms produce good assignments. All algorithms in [3]–[6] are centralized. Post *et al.* proposed in [7] a distributed version of their algorithm whereby all information upon the topology of the network and link transmission demands is distributed through the entire network. This allows each node to apply independently the centralized version of the algorithm. In [8], a distributed assignment algorithm that handles node failures has been proposed. However, this algorithm does not take into account the link traffic requirements. Other distributed algorithms that yield TDMA cycles have been proposed in [9]–[10]. Additional studies [11]–[12] have focused on applying random access schemes to the multihop packet radio network.

In this work, we focus on the model where primary and secondary conflicts are not permitted. In this case, the problem of determining the optimal assignment is known to be NP-complete [1]. Our approach is to distributedly and dynamically assign transmissions in each slot, in order to achieve a high slot utilization. The key issue is that the information kept by the individual node is limited and dynamically adapted to topology and traffic changes. In Section II, we present the underlying model of the network under consideration, state the assignment problems, and present centralized algorithms for solving them. In Section III, we demonstrate the distributed implementation of these algorithms. In Section IV, we discuss the issue of a long-term channel assignment. We present two

Manuscript received March 10, 1987; revised January 26, 1988. M. Sidi was supported in part by the Bat-Sheva de Rothschild Fund for Advancement of Science and Technology.

I. Cidon is with the IBM Thomas J. Watson Research Center, Yorktown Heights, NY 10598.

M. Sidi is with the Department of Electrical Engineering, Technion-Israel Institute of Technology, Haifi 32000, Israel. This work was done while he was at IBM T. J. Watson Research Center, Yorktown Heights, NY 10598.

IEEE Log Number 8930164.

distributed algorithms that eventually produce a TDMA cycle. Finally, in Section V we give performance bounds for the assignment algorithms along with some analytical and simulation results.

II. Network Model and Slot Assignment Problems

In this section, we first describe the network model that is used. Then we describe the various versions of the slot assignment problem that can be considered. We indicate that some versions are NP-complete problems, while the centralized solutions of other versions have polynomial complexity.

A. Network Model

Consider a multihop packet radio network where all nodes communicate through a common wide-band transmission channel. A symmetric connectivity pattern is assumed, i.e., if node i hears node j then node j also hears node i and we call such two nodes *neighbors*. We also say that i and j are one hop away from each other. Time is slotted and transmissions of packets are synchronized and take exactly one transmission slot. At any slot, a node is said to be *ready* if it has a packet ready for transmission at the beginning of the slot (otherwise it is said to be *idle*). The *intended receiver* is the node to which the packet is destined. If two or more nodes transmit packets simultaneously, then a node that hears both these transmissions cannot receive any of them successfully (there is a conflict at that node). In addition, a transmitting node is unable to receive a packet successfully.

The nodes of the network are assumed to have distinct priorities. Without loss of generality, we assume that node i has priority i. Obviously, the priority among nodes can be fixed (remain constant in time), or can be changed according to some rule such as round robin or random order [2]. This issue will be further discussed in Section IV.

B. Maximum and Maximal Slot Assignment

A slot assignment problem is the problem of selecting a set of ready nodes that will transmit concurrently during a given transmission slot. Various slot assignment problems can be formulated depending on the available information (for instance, whether the intended receivers are known or not known), and on the required solutions.

In the following, we formulate two similar *maximum slot assignments* problems that differ in the available information.

Maximum Slot Assignment—Problem 1: Let the network topology and the set of ready nodes at the beginning of a slot be given. Assign those ready nodes that will transmit in the current slot, so that the number of successful transmissions will be *maximum* (and thus maximize the utilization of the slot).

Maximum Slot Assignment—Problem 2: Let the network topology, the set of ready nodes, and the corresponding intended receivers at the beginning of a slot be given. Assign those ready nodes that will transmit in the current slot, so that the number of successful transmissions will be *maximum* (and thus maximize the utilization of the slot).

It is obvious that a solution of Problem 1 yields a conflict-free transmission in the sense that no two (or more) nodes that are one-hop away or two hops[1] away (two-hop neighbors) from each other will be assigned to transmit during the same slot. In a solution to Problem 2, nodes that are one or two hops away from each other can be assigned to transmit during the same slot, if there is no conflict at any of the intended receivers.

The maximum assignment problems turn out to be very hard combinatorial problems, whose solutions require exhaustive search of all possibilities (exponential number of computations). The reason is that the decision problem of whether there exists a set of k (k is given) ready nodes, all of which will transmit successfully, is equivalent to the *maximum clique* problems [1] and therefore is an NP-complete problem.

Had we to solve the maximum assignment problems only once, the computational effort for doing that might have been justified, but their solutions for every slot are clearly infeasible. Moreover, such solutions require global knowledge of the status of the network (topology, sets of ready nodes, and intended receivers). Collecting such information to a central site might be very inefficient, cost-wise, and might take long times.

Consequently, we are led to seeking simpler problems whose solutions might be implemented distributedly, meaning that each ready node will be able to decide whether to transmit or not based on locally available information. In addition, these problems should yield solutions that require a reasonable amount of computations and have high channel utilization. Such are the *maximal slot assignment* problems.

Maximal Slot Assignment—Problem 3: Let the network topology and the set of ready nodes at the beginning of a slot be given. Assign those ready nodes that will transmit in the current slot, so that the number of successful transmissions will be *maximal*. By maximal we mean that no additional ready node can transmit successfully without interfering with the transmission of any of the ready nodes that are assigned and all assigned nodes will transmit successfully.

As in Problem 2 we can define a Maximal Slot Assignment—Problem 4 when the corresponding intended receivers are also given.

The maximal slot assignment problems are simpler than the maximum slot assignment problems and they can be solved in polynomial number of computations. In the following, we describe simple centralized algorithms that provide solutions for these problems. The main result of this paper is their distributed implementation in a multihop environment.

Algorithm for Maximal Assignment—Problem 3:

Let the topology and the set R of all ready nodes at the beginning of a slot be given. Then the following is an algorithm for solving Problem 3.

1. Pick (at random or according to some priority rule) a node from R. Call it i. This node will transmit in the current slot and is deleted from R.
2. Delete from R all nodes that are one and two hops away from i.
3. If R is empty then stop. Else go to 1.

[1] Nodes i and j are two hops away from each other if they are not neighbors and there is a node l that is a neighbor of both i and j.

Algorithm for Maximal Assignment—Problem 4:

Let the topology, the set R of all ready nodes, and the corresponding intended receivers at the beginning of a slot be given. Then the following is an algorithm for solving problem 4.

1. Pick (at random or according to some priority rule) a node from R. Call it i. This node will transmit in the current slot and is deleted from R.
2. Delete from R the intended receiver of i.
3. Delete from R all ready nodes that are one hop away from the intended receiver of i.
4. Delete from R all nodes whose intended receiver is i.
5. Delete from R all nodes whose intended receivers are one hop away from i.
6. If R is empty then stop. Else go to 1.

As the idea behind the two algorithms is similar and since the latter algorithm is more involved, we discuss here only the second algorithm. At the first step of the algorithm a node i is assigned to transmit in the slot. Steps 2 and 3 ensure that its transmission is successful. Steps 4 and 5 ensure that future assignments for this slot will be successful. This procedure continues until R becomes empty.

It is clear that a maximal slot assignment is achieved when the algorithm terminates. The reason is that only ready nodes whose transmissions cannot be successful or whose transmission will interfere with the transmission of already assigned nodes are deleted from R. The algorithm terminates when R is empty, so that no additional ready node can transmit successfully without interfering with the transmission of ready nodes that have been assigned. In addition, all the assigned nodes will transmit successfully. It is also obvious that the complexity of both algorithms is linear (number of operations is proportional to the number of nodes in the network).

These algorithms are essentially centralized algorithms, since their input is the topology of the network, the set of ready nodes, and the corresponding intended receivers. It is not clear at this point how to implement the algorithms in a distributed manner, since at any instant, no site within the network contains the required information.

In the following, we give a detailed description of distributed implementation of the above algorithms. To that end, we first present the structure of the control channel that is used.

The Control Channel: The control channel is used for exchanging control information and signals between neighbors. This channel may be either a part of the information channel (over which nodes are transmitting their packets), or a separate dedicated channel. The general structure of the control channel is depicted in Fig. 1. For each slot of the information channel, there are two corresponding segments in the control channel called the request segment and the confirmation segment. The request and confirmation segments are of durations t_r and t_c, respectively. Each of them is divided into N (N is the number of nodes in the network) minislots, so that each node has its own dedicated minislot. Without loss of generality, we assume that minislot i is dedicated to node i. We use $t_r(i)$ and $t_c(i)$ to denote the ith minislot of the request and the confirmation segments, respectively. Each minislot of the confirmation segment is divided into two parts, the first is denoted by $t_c^1(i)$ and the second by $t_c^2(i)$. All nodes are assumed to be synchronized.

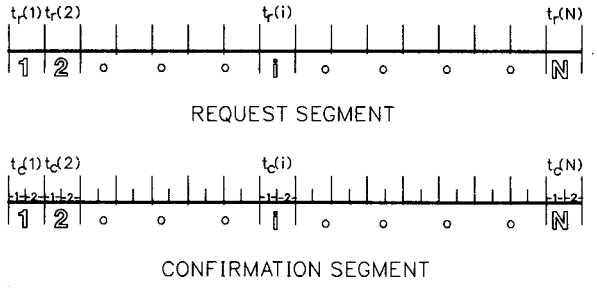


Fig. 1. The control channel.

Each node hears the activity of its *neighbors* during t_r and t_c, so that each node senses whether its neighbor is transmitting or not during the neighbor's corresponding minislot. Moreover, if two or more of its neighbors transmit in the same minislot, the node detects that activity, without knowing which and how many of its neighbors were transmitting.

III. Distributed Maximal Slot Assignment Algorithms

In the following, we present algorithms that distributedly implement the maximal slot assignment algorithms presented in Section II. These algorithms exploit the structure of the control channel to enable their distributed operation.

(A1) Algorithim for a node i—Problem 3:

⟨1⟩ If node i is ready at the beginning of the slot, it transmits a request signal during $t_r(i)$.
⟨2⟩ If node i is ready and it does not hear a deletion signal during $t_c^1(i)$, it transmits a confirmation signal during $t_c^2(i)$ and transmits its packet during the information slot. In addition, it will transmit a deletion signal during $t_c^1(l)$ for any neighbor l for which $l > i$.
⟨3⟩ If node i is ready and it hears a deletion signal during $t_c^1(i)$, then it becomes idle (in the sense that it will not transmit during the current information slot).
⟨4⟩ If node i is a neighbor of a node j that is transmitting a confirmation signal [during $t_c^2(j)$], it will transmit deletion signals during $t_c^1(l)$ for each neighbor $l > j$. In addition, if node i is ready, it becomes idle.

(A2) Algorithm for a node i—Problem 4:

⟨1⟩ If node i is ready at the beginning of the slot, it transmits the identity of its intended receiver during $t_r(i)$.
⟨2⟩ If node i is ready and it does not hear a deletion signal during $t_c^1(i)$, it transmits a confirmation signal during $t_c^2(i)$ and transmits its packet during the information slot. In addition, it will transmit a deletion signal during $t_c^1(l)$ if l is the intended receiver of i or if i is the intended receiver of l as long as $l > i$.
⟨3⟩ If node i is ready and it hears a deletion signal during $t_c^1(i)$, then it becomes idle (in the sense that it will not transmit during the current information slot).

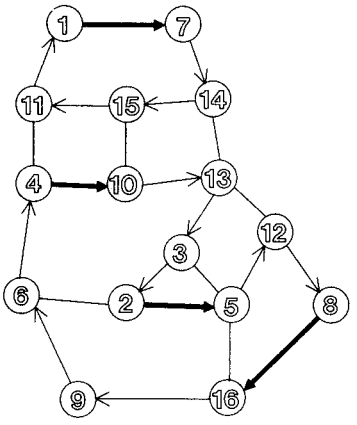


Fig. 2. A network example.

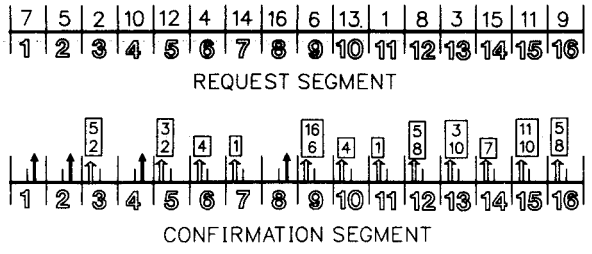


Fig. 3. Maximal link allocation example.

⟨4⟩ If node i is an intended receiver of a node j that is transmitting a confirmation signal [during $t_c^2(j)$], it will transmit deletion signals during $t_c^1(l)$ for each neighbor $l > j$.

⟨5⟩ If node i is a neighbor of a node j that is transmitting a confirmation signal [during $t_c^2(j)$], it will transmit a deletion signal during $t_c^1(l)$ if i is the intended receiver of l and $l > j$.

Note that the deletion signal sent by i to l by rule ⟨2⟩ in (A2) can be avoided since $l > i$ and it can detect directly that it should become idle. (Since either its packet is directed to a node that has been assigned to transmit or it is the intended receiver of such a node.)

The two algorithms are similar. Since the first algorithm is simpler (no identities are transmitted by the nodes), we restrict ourselves to the more involved algorithm that yields a solution to Problem 4. For this algorithm it is clear that at the end of t_r each intended receiver of every node is aware that it is the intended receiver of that node. In ⟨2⟩ and ⟨3⟩, a ready node i determines whether to transmit or not in the current slot, and if it decides to transmit, it will inhibit its neighbors, for which it is the intended receiver, from transmitting. In ⟨4⟩, node i ensures that the transmission intended to itself will be successful and in ⟨5⟩ it ensures that none of its neighbors will transmit, in order to avoid unsuccessful transmissions.

An Example—Problem 4: Consider the 16 node network depicted in Fig. 2. A link between i and j indicates that they are neighbors. An arrow from i to j indicated that i is a ready node with intended receiver j. Consequently, during the request segment, each ready node transmits the identity of its intended receiver in its corresponding minislot (⟨1⟩), as depicted in Fig. 3 (upper part). In the following, we refer to the lower part of Fig. 3 and describe the events in each minislot of the confirmation segment.

$t_c(1)$ – Node 1 does not detect a deletion signal, it transmits a confirmation signal (bold arrow) that is heard by nodes 7 and 11 (⟨2⟩).

$t_c(2)$ – Node 2 does not detect a deletion signal, it transmits a confirmation signal (bold arrow) that is heard by nodes 3, 5, and 6 (⟨2⟩).

$t_c(3)$ – Node 3 becomes idle because its intended receiver is node 2 (⟨2⟩), and the deletion signal sent by node 5 (⟨4⟩).

$t_c(4)$ – Node 4 does not detect a deletion signal, it transmits a confirmation signal (bold arrow) that is heard by nodes 6, 10, and 11 (⟨2⟩).

$t_c(5)$ – Node 5 becomes idle because it is the intended receiver of node 2 (⟨2⟩), and the deletion signal sent by node 3 (⟨5⟩).

$t_c(6)$ – Node 6 becomes idle because its intended receiver is node 4 (⟨2⟩).

$t_c(7)$ – Node 7 becomes idle because it is the intended receiver of node 1 (⟨2⟩).

$t_c(8)$ – Node 8 does not detect a deletion signal, it transmits a confirmation signal (bold arrow) that is heard by nodes 12 and 16 (⟨2⟩).

$t_c(9)$ – Node 9 becomes idle because of the deletion signal sent by nodes 16 (⟨4⟩) and 6 (⟨5⟩).

$t_c(10)$ – Node 10 becomes idle because it is the intended receiver of node 4 (⟨4⟩).

$t_c(11)$ – Node 11 becomes idle because its intended receiver is node 1 (⟨5⟩).

$t_c(12)$ – Node 12 becomes idle because its intended receiver is node 8 (⟨5⟩) and the deletion signal sent by node 5 (⟨4⟩).

$t_c(13)$ – Node 13 becomes idle because of the deletion signal sent by nodes 10 (⟨4⟩) and 3 (⟨5⟩).

$t_c(14)$ – Node 14 becomes idle because of the deletion signal sent by node 7 (⟨4⟩).

$t_c(15)$ – Node 15 becomes idle because of the deletion signal sent by nodes 10 (⟨4⟩) and 11 (⟨5⟩).

$t_c(16)$ – Node 16 becomes idle because it is the intended receiver of node 8 (⟨4⟩) and the deletion signal sent by node 5 (⟨4⟩).

Note that though node 5 is also a neighbor of node 16 (the intended receiver of node 8 that is assigned to transmit), node 16 is unable to transmit a deletion signal during $t_c(5)$ to node 5 because it is aware of the fact that node 8 will transmit only after $t_c(8)$. The same is true for node 12 that is a neighbor of node 8 and the intended receiver of node 5.

The nodes that are assigned to transmit their data packets in this example are 1, 2, 4, and 8, as indicated by the bold arrows in Fig. 2. Four concurrent successful transmissions are observed in this example. It is not too difficult to see that this slot assignment is maximal.

Validation of the Distributed Maximal Slot Assignment Algorithm—Problem 4: To validate the proposed distributed

algorithm we need to show that 1) each ready node that is assigned to transmit, does it successfully; 2) the resulting assignment is maximal. We do that by proving the following theorems:

Theorem 1: If a ready node i is assigned to transmit, then it transmits successfully.

Proof: Assume the contrary, i.e., node i is assigned to transmit but its transmission is interfered. Then either a) its intended receiver, call it j, is also assigned to transmit, or b) the intended receiver j is not assigned but at least one neighbor of j (other than i), call it l ($l \neq i$), is also assigned to transmit.

Assume that a) holds. If $i < j$, then $\langle 2 \rangle$ and $\langle 3 \rangle$ imply that j will become idle and will not be assigned to transmit, contradicting a). If $i > j$, then $\langle 2 \rangle$ implies that i heard a deletion signal during $t_c^1(i)$ and therefore (see $\langle 3 \rangle$) could not have been assigned for this slot, hence contradiction. Thus, a) cannot hold.

Assume that b) holds. If $i < l$, then $\langle 4 \rangle$ implies that node l will hear a deletion signal during $t_c^1(l)$ (transmitted by node j), contradicting b). If $i > l$, then $\langle 5 \rangle$ implies that node i heard a deletion signal (transmitted by node j) during $t_c^1(i)$ and therefore ($\langle 3 \rangle$) could not have been assigned for this slot. Thus, b) cannot hold too.

Theorem 2: The resulting assignment in a slot is maximal.

Proof: Assume the contrary, i.e., there is an additional ready node i that can transmit successfully during the current slot without interfering with the transmission of any other assigned node. Since node i was ready at the beginning of the slot, and it has not been assigned to transmit during this slot, then it must have heard a deletion signal during $t_c^1(i)$ ($\langle 2 \rangle$, $\langle 3 \rangle$). This implies that either node i was an intended receiver of some node j ($\langle 2 \rangle$) and therefore could not have been assigned, or node i is a neighbor of some node that is an intended receiver of a node that has been already assigned ($\langle 4 \rangle$), or the intended receiver of node i is a neighbor of a node that has already been assigned to transmit ($\langle 5 \rangle$). Hence, contradiction.

IV. Long-Term Assignment

In the previous section, we described distributed algorithms that yield maximal assignments per individual slots. However, treating the assignment problem independently from slot to slot has two main drawbacks. First, the assignments that are produced may cause some nodes to get permission to transmit more frequently than other nodes, while some nodes may consistently be inhibited from transmitting. Such phenomena (that is common in systems with shared resources and is usually referred to as a starvation or a fairness problem) may be due to a fixed priority (order) in which transmission requests are handled, or to the fact that the requirements from each assignment are that it would be a maximal assignment. The second drawback is that the overhead (the control portion involved in each slot in the assignment algorithm) is not negligible. Hence, if this overhead could have been saved in some slots (by reusing previous assignments, for instance) it would be helpful.

In this section, we address the issues of fairness and reducing the overhead by devising long-term assignment algorithms that assign nodes to transmit in a given slot by taking into account the fairness issue. Two long-term assignment algorithms are introduced. The first, called the Round Robin Algorithm (RRA), solves the fairness problem and preserves the property of maximal assignment in each slot. It can also be reused for more than a single cycle in order to reduce the overhead. The second, called the Wait-For-Neighbors Algorithm (WFNA), does not preserve the property of maximal assignment in each slot, but surprisingly if the set of ready nodes is kept fixed, it produces a TDMA cycle in which every node in the network is assigned to transmit an equal number of slots.

Note that the fairness and the overhead problems can be decoupled. The fairness problem is solved by devising assignment algorithms that favor nodes which were not assigned in previous slots. This operation creates fair scheduling periods. The overhead problem is solved by repeating the same fair scheduling patterns for several times. The maximal slot assignment algorithm (A1) serves as a basic building block for both long-term assignment algorithms.

The Round Robin Algorithm (RRA): In this algorithm, the maximal slot assignment algorithm (A1) is performed in each slot. In order to give each transmitter a chance for transmission, the priority (order) of the allocation of the minislots is cycled. The last node of the previous slot becomes the first node of the current slot. All other nodes are shifted one minislot [2]. A TDMA-RR is defined as a complete round robin cycle. Let N be the maximal number of transmitters in the network and assume that all nodes have packets to transmit. The RRA guarantees that in any cycle of N slots, each node transmits at least once, and the assignment in each slot is maximal.

The RRA resolves the fairness issue by a complete rotation of the access priority among the nodes. In addition, it preserves the efficiency of the maximal slot assignment algorithm. Note that the round robin rotation among nodes only changes the order in which the algorithm schedules nodes for transmission for every slot. However, any node may still transmit at any slot provided that no collision occurs with the transmission of higher priority nodes. The scheme is adaptive in the sense that slots are not firmly reserved. If high-priority nodes do not transmit, then low-priority nodes are assigned to transmit. The algorithm schedules transmissions according to the load demand. For example, if node i always has packets to transmit while all its one-hop and two-hop neighbors have no packets, then i will transmit during all slots. This makes the RRA especially attractive for networks with heterogeneous traffic requirements.

One implication of the RRA is the possibility to perform the algorithm less frequently. When the RRA is activated and a TDMA-RR is produced, the same TDMA cycle (of N slots) can be used until the algorithm is reactivated to produce a new TDMA-RR cycle, thus saving the overhead incurred in determining the maximal assignment in each slot. (Note that some of the nodes will transmit more than once during the TDMA cycle.) The period of time between consecutive activations of the RRA will depend on the dynamic behavior of the network, i.e., how fast the network traffic requirements and topology are changed.

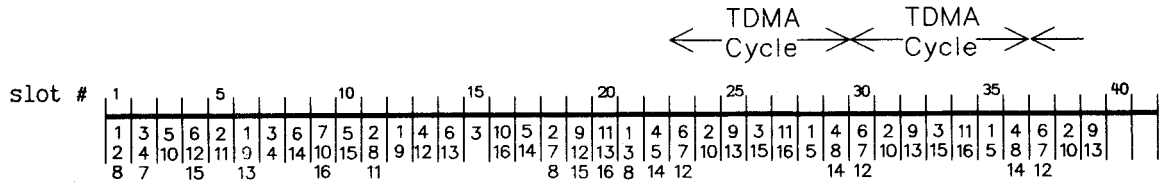


Fig. 4. Wait-for-neighbors example.

The Wait-For-Neighbors Algorithm (WFNA): In this algorithm, the basic maximal slot assignment algorithm (A1) is performed in each slot and during its operation the priority among the nodes remains fixed (i.e., each node always uses the same minislot in the control portion of the channel). However, whenever a node is assigned to transmit in a slot, it does not participate in the slot assignment algorithm in the following slots, until it hears that *all* its *ready* neighbors have been assigned to transmit at least once, and only then it resumes its participation in the slot assignment algorithm. This is similar to the edge-reversal operation presented in [14] and [15] and to the algorithm described in [16].

An example of the operation of the WFNA for the network of Fig. 2 is depicted in Fig. 4. It is assumed that all nodes have packets to transmit. In the first slot, the basic algorithm (A1) is performed and nodes 1, 2, and 8 are assigned to this slot. From this point, node 1 will not reattempt to transmit until it hears that all its neighbors (7, 11) have been assigned a slot (hence, it reattempts and succeeds in slot 6). Similarly, node 2 waits for nodes 3, 5, and 6 and then (in slot 5) reattempts and succeeds. Node 8 waits for nodes 12 and 16 and hence reattempts in slot 10 (unsuccessfully) and tries again in slot 11 (now successfully). In slot 2, all nodes (except nodes 1, 2, and 8) perform the algorithm A1 and nodes 3, 4, and 7 are assigned to this slot. Node 3 will wait for 2, 5, and 13, node 4 will wait for 6, 10, and 11, and node 7 will wait for 1 and 14. Hence, node 3 starts to participate in the algorithm again in slot 7, node 4 starts to participate in the algorithm again in slot 6 (but succeeds only in slot 7), and node 7 starts to participate in the algorithm again in slot 9. The example thus continues and we see that in slot 29 the algorithm converges to a TDMA cycle of seven slots (utilization of 16/7 nodes per slot) during which every node transmits exactly once.

Note that the WFNA does not preserve the property of maximal assignment in each slot, because of the additional restriction upon the nodes to wait for their ready neighbors to be assigned at least once before asking for a new transmission. The main feature of the WFNA is that under heavy load (all nodes always have packets to transmit) it produces a TDMA cycle. During this cycle, all nodes are assigned exactly the same number of slots. This is stated and proved in the sequel.

Theorem 3: Assume that the network is connected and all nodes have packets to transmit. Then the Wait-For-Neighbors Algorithm converges to a single TDMA cycle in a finite number of iterations. Furthermore, during this TDMA cycle, each node is assigned at least once and all nodes are assigned the same number of slots.

In order to prove Theorem 3, the following lemmas are used.

Lemma 1: At least one node is assigned for transmission in each slot when the WFNA is employed.

Proof: If there exists some ready node i that has never transmitted, then this node will request transmission (will not wait for any neighbor to transmit). Since A1 is a maximal assignment algorithm and at least one node requests transmission, at least one node will be assigned. If all nodes have transmitted at least once then let i be the node that has not been transmitting for the longest period. Since all its neighbors have transmitted after its last transmission (recall that neighbors are never transmitting in the same slot), i will request transmission.

Lemma 2: The WFNA becomes periodic after a finite number of slots.

Proof: Define the scheduling graph at slot n, SG_n, as the directed graph whose nodes represent the nodes of the network and a directed edge from node i to its neighbor j in SG_n indicates that at the beginning of slot n node i is waiting for the transmission of its neighbor j before requesting transmission. The graph SG_n is clearly an acyclic directed graph for all n and it uniquely determines the nodes that will transmit in slot n and hence the scheduling graph of slot $n + 1$, SG_{n+1}. This implies that at a given slot n all future assignments are uniquely determined by SG_n.

The number of scheduling graphs is finite and therefore the sequence of transmissions (that is uniquely determined by the initial scheduling graph) must become periodic after a finite number of slots.

Lemma 3: When the WFNA becomes periodic, all nodes are assigned to transmit the same number of times in each cycle.

Proof: We first prove the lemma for some pair of neighbors i and j. Assume that i and j transmit k_i and k_j times in a cycle, respectively. In any two consecutive cycles, i transmits $2k_i$ times. Since i waits for all its neighbors' transmission before transmitting, there is at least one transmission of j between any two transmissions of i. Thus, in any two consecutive cycles there are at least $2k_i - 1$ transmissions of j. This implies $2k_j \geq 2k_i - 1$. Similarly, $2k_i \geq 2k_j - 1$, which implies that $k_j = k_i$. Since the network is connected, if the lemma holds for all pairs of neighbors, then it holds for all nodes.

Theorem 3 follows from Lemmas 1-3.

V. Performance Bounds and Analysis

In this section, we present some performance bounds and analysis of the distributed algorithms presented in Section III. We mainly focus on the simpler algorithm (A1) that solves Problem 3. The major quantity of interest in the analysis is the

number of concurrent successful transmissions in a slot that in general depends on the network topology and on the set of ready nodes.

A. Simple Bounds

Consider some network with a fixed topology. Let R be the set of ready nodes at the beginning of a slot and let $|R|$ indicate the cardinality of the set ($|R| = N$ when all nodes of the network are ready). Let $L(R)$ be the number of concurrent successful transmissions in the slot when algorithm (A1) is applied. For each ready node i, let $D_i(R)$ be the set containing node i and all its one-hop and two-hop neighbors that are ready, and let $D(R) = \max_{i\in R}|D_i(R)|$. Then

$$L(R)\geq |R|/D(R). \tag{1}$$

The reason for (1) is that whenever a ready node is assigned to transmit, all its one-hop and two-hop neighbors cannot be assigned. Therefore, at least $|R|/D(R)$ ready nodes will be assigned in the worse case. From (1) we conclude that whenever $D(R)$ does not depend on $|R|$ (such is the case in bounded degree networks), then $L(R)$ is proportional to $|R|$.

The relation in (1) is for the worst case. More often, we are interested in the average behavior. As the number of concurrent successful transmissions depends on the priorities of the ready nodes, we denote by $\bar{L}$ the average (over all possible permutations of the priorities) number of concurrent successful transmissions. Let $D(\bar{R}) = \Sigma_{i\in R}D_i(R)/|R|$ be the average number of ready nodes that cannot be further assigned per assigned node. Then as in (1) we have

$$\bar{L}\geq |R|/D(\bar{R}). \tag{2}$$

B. Analysis

Tandem Networks: Consider a tandem network consisting of N nodes. In this network, nodes i and j are neighbors if and only if $|i - j| = 1$. Assume that at the beginning of a slot each node is ready with probability p. Let $L_m(N)$ be the average number of successful transmissions in an N-node tandem with m ready nodes. Then $L(N)$, the *average* number of successful transmissions in an N-node tandem is

$$L(N)=\sum_{m=0}^{N}\binom{N}{m}p^m(1-p)^{N-m}L_m(N). \tag{3}$$

The quantities $L_m(N)$ can be calculated via the following recursion:

$$L_0(N)=0\ \forall N \tag{4a}$$

$$L_1(N)=1\ \forall N \tag{4b}$$

$$L_m(N)=1+\frac{1}{N}\left\{2\sum_{k=0}^{\min(2,m-1)}\cdot\frac{\binom{2}{k}\binom{N-3}{m-k-1}}{\binom{N-1}{m-1}}L_{m-k-1}(N-3)\right.$$

$$+2\sum_{k=0}^{\min(3,m-1)}\cdot\frac{\binom{3}{k}\binom{N-4}{m-k-1}}{\binom{N-1}{m-1}}L_{m-k-4}(N-4)$$

$$+\sum_{i=3}^{N-2}\sum_{k=0}^{\min(4,m-1)}\sum_{j=0}^{\min(i-3,m-k-1)}\cdot\frac{\binom{4}{k}\binom{i-3}{j}\binom{N-i-2}{m-j-k-1}}{\binom{N-1}{m-1}}$$

$$\left.\cdot\,[L_j(i-3)+L_{m-j-k-1}(N-i-2)]\right\}$$

$$2\leq m\leq N \tag{4c}$$

Equations (4a) and (4b) are obvious. To understand (4c) assume that the algorithm assigns a ready node at position i of the tandem (this happens with probability $1/N$). If $i = 1$ or $i = N$ we are left with a tandem with $N - 3$ nodes where $m - k - 1$ nodes are ready (k is the number of ready nodes that are one and two hops away from i, so in this case $0 \leq k \leq 2$). The next term in (4c) corresponds to picking node 2 or $N - 1$. Finally we sum all cases that $3 \leq i \leq N - 2$. In these cases $0 \leq k \leq 4$ and after assigning node i we are left with two tandems, one with $i - 3$ nodes and the other with $N - i - 2$ nodes. Taking j to be the number of ready nodes in the tandem of $i - 3$ nodes, the other tandem will have $m - j - k - 1$ ready nodes.

When all nodes are ready ($p = 1$), let $L(N) = L_N(N)$. Then from (3) and (4) we obtain

$$L(1)=L(2)=L(3)=1 \tag{5a}$$

$$L(N)=1+\frac{1}{N}\left\{2L(N-3)+2L(N-4)+\sum_{i=3}^{N-2}[L(i-3)+L(N-i-2)]\right\}\qquad N\geq 4. \tag{5b}$$

After some algebra, the above reduces to

$$L(N+1)=(1+NL(N)+2L(N-2))/(N+1)\quad N\geq 3. \tag{6}$$

Let $G(z) = \Sigma_{N=1}^{\infty}L(N)Z^N$ and $G'(z) = dG(z)/dz$. Then from (6) we have

$$G'(z)=1/(1-z)^2+2z^2G(z)/(1-z)\text{ with }G(0)=0. \tag{7}$$

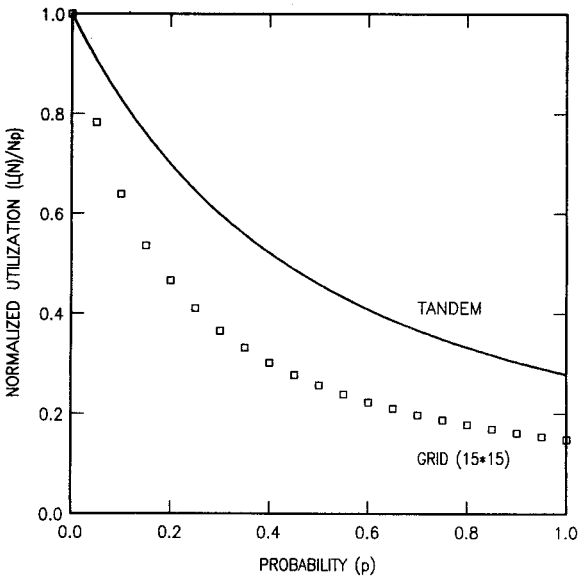


Fig. 5. Results for a tandem network and a grid network.

Solving (7) we obtain

$$G(z)=\frac{\int_0^z e^{(x-z)(x+z+2)}\,dx}{(1-z)^2}. \tag{8}$$

From (8) we can obtain the limiting behavior of $L(N)/N$ for large N by calculating $\lim_{z\to 1}(1-z)\int G(z)dz$. The results is $L(N)/N \simeq \int_0^1 e^{(x-1)(x+3)}\,dx = 0.2745$. For a tandem network the bound in (1) implies that $L(N)/N \geq 0.2$. In addition, it is easy to find the maximum assignment in a tandem network (just assign every third node), that yields $L(N)/N \simeq 1/3$. Thus, the performance of our algorithm for a tandem is just 17.6 percent below the optimal performance when $p = 1$. When p becomes smaller, it is expected that its performance will be even better when compared to the optimal assignment. The normalized utilization $(L(N)/Np)$ for a tandem with $N = 100$ nodes is depicted in Fig. 5 as a function of the probability that a node is ready—p. We also plotted in Fig. 5 simulation results for the normalized utilization of a grid network with $N = 15 \times 15 = 225$ nodes. In this grid network, each interior node has four neighbors, nodes on the boundaries have three neighbors, and the four nodes on the corners have two neighbors. Each point in Fig. 5 corresponds to 100 000 runs of the simulation. In each run, we randomly assigned a distinct identity to each node, in the range between 1 and 225. For each node we have assigned a random binary decision whether the node has a ready packet (with probability p) or not (with probability $1 - p$). The curve in Fig. 5 corresponds to the average behavior of the quantity $L(N)/NP$ when algorithm A1 is employed. It is interesting to note that when $p = 1$, the optimal performance would be obtained if nodes are assigned to transmit according to knight moves in chess. Such assignments yield utilization of 0.2 for large grids.

VI. Summary

New distributed dynamic channel assignment algorithms for a multihop packet radio network have been introduced. The algorithms ensure conflict-free transmissions by the nodes of the network. The basic idea of the algorithms is to split the shared channel into a control segment and a transmission segment. The control segment is used to avoid conflicts among nodes and to increase the utilization of the transmission segment. We have also shown how these algorithms can be used in order to determine TDMA cycles with spatial reuse of the channel. As a final remark, we would like to point out that we have presented a basic control structure. At the expense of extra signaling, one may add other mechanisms to the basic assignment algorithms that handle message priorities, variable length messages, multiple information channels, etc.

Acknowledgment

We would like to thank Prof. S. Zaks for his helpful comments and suggestions.

References

[1] E. Arikan, "Multi-access in packet-radio networks," M.Sc. thesis, Lab. Inform. Decision Syst., MIT, LIDS-TH-1234, Sept. 1982.
[2] L. Kleinrock and M. Scholl, "Packet switching in radio channels: New conflict-free multiple access schemes," *IEEE Trans. Commun.*, vol. COM-27, pp. 1015–1029, Aug. 1979.
[3] B. Hajek and G. Sasaki, "Link scheduling in polynomial time," *IEEE Trans. Inform. Theory*, to be published.
[4] R. G. Ogier, "A decomposition method for optimal link scheduling," in *Proc. 24th Annu. Allerton Conf.* Oct. 1986, pp. 822–823.
[5] R. Nelson and L. Kleinrock, "Spatial TDMA: A collision-free multihop channel access protocol," *IEEE Trans. Commun.*, vol. COM-33, pp. 934–944, Sept. 1985.
[6] M. J. Post, P. E. Sarachik, and A. S. Kershenbaum, "A 'biased greedy' algorithm for scheduling multi-hop radio networks," in *Proc. Conf. Inform. Sci. Syst.*, 1985, pp. 564–572.
[7] ——, "A distributed evolutionary algorithm for reorganizing network communications," in *Proc. IEEE MILCOM*, 1985.
[8] D. J. Baker, J. Wieselthier, and A. Ephremides, "A distributed algorithm for scheduling the activation of links in a self-organizing mobile radio network," in *Proc. IEEE Int. Conf. Commun.*, 1982, pp. 2F.6.1–5.
[9] I. Chlamtach and A. Lerner, "Fair algorithm for maximal link activation in multihop packet radio networks," *IEEE Trans. Commun.*, vol. COM-35, pp. 739–746, July 1987.
[10] I. Chlamtach and S. Pinter, "A distributed nodes organization algorithm for channel access in a multi-hop dynamic radio network," *IEEE Trans. Comput.*, vol. C-36, pp. 728–737, June 1987.
[11] R. Boorstyn and A. Kershenbaum, "Throughout analysis of multihop

packet radio," in *Proc. ICC'80*, Seattle, WA, June 1980, pp. 13.6.1–13.6.6.

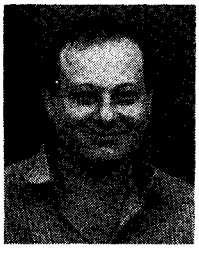
[12] F. A. Tobagi and D. H. Shur, "Performance evaluation of channel access schemes in multihop packet radio networks with regular structure by simulation," *Comput. Networks ISDN Syst.*, vol. 12, pp. 39–60, Aug. 1986.

[13] D. Bertsekas and R. Gallager, *Data Networks*. Englewood Cliffs, NJ: Prentice-Hall, 1987.

[14] E. M. Gafni and D. P. Bertsekas, "Distributed algorithms for generating loop-free routes in networks with frequently changing topology," *IEEE Trans. Commun.*, vol. COM-29, pp. 11–18, Jan. 1981.

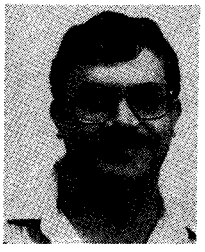
[15] V. C. Barbosa and E. M. Gafni, "Concurrency in heavily loaded neighborhood-constrained systems," preprint.

[16] J. Malka, S. Moran, and S. Zaks, "A simple distributed scheduler—Analysis and applications," Extended Abstract, 1987.



Israel Cidon (M'85) received the B.Sc. (summa cum laude) and the D.Sc. degrees from the Technion–Israel Institute of Technology, Haifa, in 1980 and 1984, respectively, both in electrical engineering.

From 1980 to 1984 he was a Teaching Assistant and a Teaching Instructor at the Technion. From 1984 to 1985 he was on the faculty of the Department of Electrical Engineering at the Technion. Since 1985 he has been with IBM Thomas J. Watson Research Center where he is currently the manager of the Integrated Network Architecture group. His current research interests are in high-speed communication networks, distributed algorithms, and packet radio networks.



Moshe Sidi (S'77–M'82–SM'87) received the B.Sc., M.Sc., and D.Sc. degrees from the Technion–Israel Institute of Technology, Haifa, in 1975, 1979, and 1982, respectively, all in electrical engineering.

From 1975 to 1981 he was a Teaching Assistant and a Teaching Instructor at the Technion in communication and data networks courses. In 1982, he joined the faculty of the Department of Electrical Engineering at the Technion. During the academic year 1983–1984 he was a Postdoctoral Associate at the Department of Electrical Engineering and Computer Science at the Massachusetts Institute of Technology, Cambridge. During 1986–1987 he was a Visiting Scientist at the IBM Thomas J. Watson Research Center, Yorktown Heights, NY. His current research interests are in queueing systems and in the area of computer communication networks.

Dr. Sidi is currently the editor for communication networks for the IEEE TRANSACTIONS ON COMMUNICATIONS.