A SPATIAL-REUSE TDMA/FDMA FOR MOBILE MULTI-HOP RADIO NETWORKS

I. Chlamtac and S. Kutten

Computer Science Department
Technion - Israel Institute of Technology
Haifa 32000, Israel

ABSTRACT

This paper deals with bandwidth allocation in multi-hop mobile radio networks. The proposed assignment of transmission rights provides collision free, deterministic access to the channel by more than one node at a time thus increasing network utilization through channel reuse. The assingment of transmission rights at each node is obtained by executing a distributed algorithm. The algorithm works for unrestricted topologies of both stationary and mobile networks. Utilizing prevailing operational assumptions the algorithm deals with assingment changes caused by network reconfigurations without the use of a centralized factor. Depending on the nature of the topological change the effect of transmission rights reassignment can be limited to the locality of that change.

1 INTRODUCTION

In most packet radio networks not all nodes are in line of sight and within range of each other [7,8,10,9]. In the resulting "multiple-hop" network the broadcasting nature of the radio becomes a dominating factor in the channel access protocol design and its performance. Specifically, it is generally assumed that every transmission reaches all nodes in line of sight of the transmitting node and multiple transmissions arriving concurrently will "collide" and be subsequently lost. It is easy to see that a channel access (transmission right) policy which does not account for these properties may rely on a random access policy with compensation for collisions obtained by retransmissions [1,4]. For many applications and networks, e.g., military networks, a more deterministic type of channel control is required [7,8]. A controlled access policy must take into account the complex dependencies between the ready nodes wishing to transmit. Optimal assignments were shown to be intractable for a variety of criteria even in the case of stationary networks [2]. Existing controlled access allocation policies rely heavily on additional mechanisms, utilizing special control channels, regional control nodes, etc [3,5,6]. In this paper we present a channel access policy based on a simple distributed algorithm executed by all active nodes in the network without centralized or regional agents. The execution of the algorithms is invoked at network initiation or following a topological (network graph) change which may result from failure or node movements. The obtained assignment of transmission rights, in units of time slots or frequencies at each node guarantees freedom from collisions, and an upper bound on the total collection of distinct units allocated by the algorithm. This bound is only the function of the number of neighboring nodes rather than the total number of network nodes and thus channel reuse can be provided.

The allocation algorithm is linear in the number of control messages and limits the effect of a local topological change on the slot assignment to a local subset (subgraph) of network nodes. The only operational assumptions made are the node's knowledge of its neighbors and the ability of each node to detect a topological change caused by one of its neighbors [5].

2 MEDIUM AND MODEL DESCRIPTION

A node wishing to transmit a message can use time and bandwidth, considered to be the shared network resources. Resource sharing is accomplished by allocation of basic units we term *phases*. E.g. if time division is used, phases correspond to time slots; using frequency division a phase corresponds to a band. Without defining exactly at this point the way phases are constructed, we make the following model related assumption which must always hold: transmissions in different phases will not collide. The collision of transmissions is defined below.

An undirected graph $G(V,E)$ represents the network with vertices in V the network nodes, and an undirected edge $e \in E$ interpreted as two anti parallel directed edges. Each directed edge $u \rightarrow v$ corresponds to the following $\rightarrow$ relation between u and v: v receives every signal transmitted by u. In other words $u \rightarrow v$ denotes that v is in line of sight and within range of u.

To ensure that v not only receives but also *correctly* interprets a transmission initiated by u, the following conditions must *also* be satisfied:

Condition 2.1: v does not transmit in the same phase u does.

Definition: A neighbor, w, of v is an *incoming neighbor* if there is an edge directed from w to v.

Condition 2.2: u is the only incoming neighbor of v transmitting in the same phase.

If condition 2.1 does not hold we say that u's transmission collided in v. If condition 2.2 does not hold, we say that the transmission of the incoming heighbors, u and w, have collided in v.

3 THE TRANSMISSION RIGHTS ALLOCATION ALGORITHM

We look for a phase allocation among all network nodes which guarantees that a transmission by a network node will be correctly received by all its neighbors.

The proposed distributed algorithm generates such an allocation for unrestricted network graphs and provides for distributed network reconfiguration. The collection of all distinct phases assigned by the algorithm is proovably bounded. If e.g. the phase is implemented by time division the resulting collection of time slots defines a cycle of constant, known length.

Since the number of phases will be generally smaller than the number of network nodes, the radio channel will be reused for simulataneous non colliding transmissions, i.e., a spatial reuse is obtained.

3.1 Description of the Phase Assingment Algorithm

The following distributed algorithm phases the network graph in such a way that each node is assigned a phase which is different from the phases of all it's neighbors, and it's neighbors' neighbors. Thus phasing the network's graph corresponds to the coloring of a graph whose nodes are the network's nodes and it's set of edges, E, includes an edge between every two neighbors in the network, and between every two nodes having a common neighbor in the network. We term that graph the supergraph of the network's graph.

The algorithm is based on the following centralized graph coloring greedy algorithm: number the colors, then repeat the following. Choose an uncolored node, i, and assign it the color having the smallest number among the colors which are not already forbidden by the following restriction: when acting on the supergraph the assigned color is not yet assigned to a neighbor of i. Thus, when acting on the network graph the assigned color must be one which is not yet assigned to a neighbor of i, and to a neighbor of i's neighbor. Now let K be the maximum number of neighbors of a node in the network's graph. Then K^2 is the maximum number of neighbors a node in the supergraph may have and K^2+1 is the upper bound on the number of colors assigned by the algorithm coloring the supergraph (and thus the upper bound on the number of phases assigned to the network's graph nodes).

We now informally describe the distributed algorithm. (In the following simplified description several redundant messages may be sent without changing the Order of the number of messages. A way to avoid the redundant messages will be described later.

We assume that the network nodes have distinct, fully ordered identities (IDs). Node's ID is known to the node itself in (LOCAL.ID) and to its neighbors (in their ListOfNeighbors- which is ordered from the highest neighbor's ID to the lowest). Any nonempty subset of nodes can initiate the algorithm. A node enters the algorithm either on receiving the message Wake caused by an upper layer algorithm (protocol) or by the first reception of a message sent by another node executing the algorithm. On entering the algorithm each node, i, needs the permission of all it's neighbors and all it's neighbors' neighbors in order to assign itself a phase. This is done in order to simulate the choosing of the next node by the centralized graph coloring algorithm: whenever a node, i, may assigns itself a phase, no neighbor of i, or of i's neighbor, is allowed to assign itself a phase till it receives i's permission. Neighbors' permission is accepted directly from the neighbors. The neighbors' neighbors' permission is accepted through the neighbors. Permissions are passed in messages together with a set of RESTRICTIONS specifying phases a node will not assign itself. Node i keeps the phases it will assign itself in LOCAL.RESTRICTIONS, initially empty. Each time i receives a permission, together with RESTRICTIONS it adds the RESTRICTIONS to LOCAL.RESTRICTIONS. Node i also maintains LOCAL.OneHopRestrictions initially empty. Restrictions will also be added to LOCAL.OneHopRestrictions only when they are restrictions caused by neighbors, and not by neighbor's neighbors. In order to achieve this selective addition a node distinguishes between the source of the restrictions: The node will send (1) neighbors' phases (LOCAL.OneHopRestrictions) and (2) its own phase (LOCAL.Phase) If when sending a permission the node's own phase is not yet assigned, the node will send a DummyPhase.

Two types of messages are used by nodes to send permissions to their neighbors: TwoHopsPermit and OneHopPermit. Node i sends TwoHopsPermit to node j if node j is the largest (identity) among i's neighbors (including i itself) not yet having a phase. OneHopPermit is sent by node i (to all neighbors others than those to which TwoHopsPermit has been sent) when it has assigned itself a phase. This frees its neighbors to send TwoHopsPermit to other nodes. As explained above both messages are sent together with LOCAL.OneHopRestrictions, LOCAL.PHASE and LOCAL.ID.

We now describe the actions of a node participating in the algorithm. On waking node i sets to "true" the variable NodeAwake and sends Wake to its neighbors. (We assume that only the first such message will generate the neighbor's Wake event.) If the identity of node i (ID) is not larger than the ID-s of all its neighbors, node i sends the largest identity neighbor, j, a TwoHopsPermit message. (Notice that at this point the set of Restrictions is still empty.) If j receives TwoHopsPermit with no restrictions from all its neighbors than clearly j's identity is larger than the identity of all it's neighbors' neighbors. In general a node which has received TwoHopsPermit(RESTRICTIONS) from all its neighbors can choose a phase number not included in the accumulated restrictions. Having

eventually chosen a phase (its LOCAL.PHASE), node i sends OneHopPermit to all neighbors except the next largest neighbor (among neighbor's not yet having a phase) which is sent a TwoHopsPermit). Together with the permits node i sends its own LOCAL.PHASE, and from its OneHopRestrictions. At i's neighbor i's LOCAL.PHASE becomes part of OneHopsRestrictions and i's OneHopRestrictions become LOCAL.Restrictions.

3.2 Phases Assignment Algorithm

```
Type ToWhomPermit=(MYSELF,OTHER,NONE);
Function NextPermit:ToWhomPermit;
begin
  if HEAD(ListOfNeighbors) > LOCAL.ID
    then NextPermit:=OTHER;
    (*HEAD(list) returns the value of the first item (the highest ID) in the list
    (without removing the item from the list*)
  else begin
    if |ForbiddingNeighbors|=0 then nextPermit:=
                                MYSELF;
    (*the set of ForbiddingNeighbors contains ID's of
     those neighbors from which a TwoHopsPermit
     has not yet been received*)
    else NextPermit:=NONE;
  end
end;

Procedure SendPermit;
begin
  v:=RemoveFirstIn(ListOfNeighbors);
  (* the highest ID is removed from
    the list and put in v*)
  send TwoHopsPermit(
      LOCAL.SetofRestrictions,LOCAL.PHASE,LOCAL.I
    to v;
end;

Procedure AssignPhase;
begin
  LOCAL.PHASE:=
    MIN(SetOfAllPositiveIntegers-
    LOCAL.RESTRICTIONS);
  send Permit;
  for each v in ListOfNeighbors do
    send OneHopPermit(
      LOCAL.RESTRICTIONS,LOCAL.PHASE,LOCAL.ID)
      to v;
end;

on(* reception of *) Wake
begin
  LOCAL.NodeAwake:=true;
  (w1) LOCAL.PHASE:=DummyPhase;
  (w2) if NextPermit=MYSELF then AssignPhase
  (w3) else if NextPermit=OTHER then SendPermit;
  (w4) else (*NextPermit=NONE*)
  (w5) for each u in LOCAL.ListOfNeighbors-v
       do concurrently
         send (*message*) Wake to u;
  wait (*for a message*);
```

```
end;

on (*reception of*)
  TwoHopsPermit(RESTRICTIONS,PHASE,NODE)
begin
  if not LOCAL.NodeAwake then perform Wake
        excluding statements w2 to w4;
  LOCAL.RESTRICTIONS:=LOCAL.RESTRICTIONS+
                    RESTRICTIONS;
  OneHopRestrictions:=OneHopRestrictions+{PHASE}
  ForbiddingNeighbors:=ForbiddingNeighbors-
{NODE};
  if NextPermit=MYSELF then AssignPhase
  else if NextPermit=OTHER then SendPermit;
end;

on (*reception of*)
   OneHopPermit(RESTRICTIONS,PHASE,NODE)
begin
  OneHopRestrictions:=OneHopRestrictions+{PHASE};
  LOCAL.RESTRICTIONS:=LOCAL.RESTRICTIONS+
                    RESTRICTIONS+{PHASE};
  if NextPermit=MYSELF then AssignPhase
  else if NextPermit=OTHER then SendPermit;
  if |ListOfNeighbors|=0 then Terminate;
end;
```

3.3 Properties of the Phases Assignment Algorithm

Definition: Let N(i) be the set of i's neighbors and neighbors' neighbors.

Lemma 1: When node i assign itself a phase, no node j in N(i) has a TwoHopsPermit from all nodes in N(j).

Proof: We prove by induction on the number of nodes in the graph. If n=1 then the lemma is trivial. Assume the lemma holds for every graph containing less than n nodes and let G(V,E) contain n nodes. Let i be the node in V with the lowest identity. Let G' be G with the following changes. Remove i from G and connect with an edge every two neighbors of i. For G' the lemma holds by the induction hypothesis. If i is not removed (and the new connecting edges are not added) then MYSELF must be the value given by NextPermit only in it's last call (by i) as i's identity is always smaller than HEAD(ListOfNeighbors). Thus the algorithm on G works as on G' till i has sent TwoHopsPermit to all it's neighbors. Thus and by the need for i to receive TwoHopsPermit from all neighbors- the lemma holds.

Lemma 2: Eventually all nodes are assigned phases.

Proof: The proof is similar to the proof of lemma 1.

THEOREM 1: Eventually all nodes are assigned phases which meet the restrictions and the number of phases is bounded by K^2.

Proof: The theorem follows from the lemmas, from the centralized graph coloring algorithm described earlier and from the use of the assembled RESTRICTIONS in the phase assignment process.

THEOREM 2: No more than 6|E| message are sent by the algorithm.

Proof: Each node sends over each link one Wake message, at most one OneHopPermit, and one TwoHopsPermit.

Observation: The Wake message from a node i to node j from which i has already received a message is redundant (see section 3.1).

THEOREM 3: The computation time complexity of the algorithm is O(|E|).

Proof: By noticing that a constant number of computation steps is performed per each message.

3.4 Adapting to Topological Changes

When topological changes occur,. changes in the phases assignments may be necessary. We assume that a node whose set of neighbors has been changed is aware of such a change [5]. One way to achieve the required change in the phase assignment is to cause a Wake event in every node which notices a change in its set of neighbors and to execute the algorithm as described in 3.2. However, the changes in phase assignment can often be restricted to a locality smaller than the whole network saving control messages, computation time and possibly reconfiguration time as follows.

In this discussion we shall assume that the changes are handled serially s.t., the update process for a given change is started only after the previous update has been terminated.

Definition: For any $V' \subseteq V$ define a *bordering set* of V', Bs(V') as a set of nodes for which the following holds. Every path from a node in V' to a node not in $V' \cup Bs(V')$ passes through a node in Bs(V').

Let G be a network graph for which the Phases assignment algorithm has been invoked and terminated and in which a topological change has occurred. Let V' be the set of nodes neighboring the topological change. Assume that in a new invokation of the algorithm V' has a bordering set s.t. every node in the bordering set has a LOCAL.RESTRICTIONS which is included in the LOCAL.RESTRICTION set generated at the previous invocation. Clearly every node not in $V' \cup Bs(V')$ can be assigned the same phase as the one assined to it before the new invocation. Based on this observation we now can define an update procedure to deal with topological changes.

Instead of a Wake message being sent by every waking node, Wake messages will be sent only in two cases:

(1) By a node which has received it's first TowHopsPermit.

(2) By a node which has been awaken by an upper layer protocol (and not by a message) and it's identity is larger than the identity of all it's neighbors.

This change in the waking policy enables nodes to stop the coloring process from covering the whole network. When a node assigns itself a phase and finds out that its new set of LOCAL.RESTRICTIONS is included in the old set, it adds the field CANCEL to the permission messages it sends to its neighbors. Let j be a node which has been awaken by a message and not by an upper layer protocol. Let W(j) (clearly $W(j) \subseteq V' \cup Bs(V')$), be the set of j's neighbors which have sent it the message Wake. When j receives Permissions with the field CANCEL from all the nodes in W(j), j still does not know whether a neighbor k not in W(j), has not sent a message, or k has sent a message which has not yet arrived. Thus, j sends messages to all its neighbors not in W(j) asking for a REPLY. (Notice that if k did send a Wake message, then the Wake message must have been received at j before the REPLY message.) When j has received a REPLY from all neighbors not in W(j) (and a CANCEL from all neighbors in W(j)) the algorithm in j will terminate.

Notice that the update procedure can also be used to generate the initial phase assignment, although at a lower concurrency level. This observation well corresponds with the intuition which views network initialization as an instance of a topological update.

The construction of an example in which, following a topological change, only two neighbors and their neighbors will be awaken before termination is trivial.

Finally we note that it would have been nice if we could maximize the number of local maximum IDs in the network, as it would have made the execution of the algorithm more parallel. However it is easy to see that this problem is NPH (by a reduction from the Independent Set problem [10]). As an heuristic we may suggest to use the nodes phased by 1 in previous algorithm invokation, as candidates for local maximum (and their ID will thus become the less significant field in the lexicographical ordering of the nodes).

SUMMARY

We have proposed a "phase allocation" algorithm for assigning channel access rights in multi-hop, mobile radio networks. The algorithm is totally distributed and adaptive and provides each node with collision free channel access.

The phase allocation produced by the algorithm can be implemented by techniques of time division, frequency division or a combination of both. The phase allocation process is completely distributed and can be initiated by any node or a set of nodes as needed. Furthermore, it has the potential of localizing topological changes. Network protocols based on the resulting TDMA or FDMA protocols can provide therefore predictable service in both stationary and mobile radio networks.

REFERENCES

[1] Abramson, N., "The ALOHA System" *Computer Communication Networks* edited by Abramson, N. and Kuo, F., pp. 501-518, Englewood Cliffs, N.J., Prentice-Hall, 1973.

[2] Chlamtac, I. and Kutten, S., "On Broadcasting and Spatial Reuse in Radio Networks - Problem Analysis and Design of Protocols with Provable Properties", TR#273, Dept. of Computer Science, Technion - Israel Institute of Technology, Haifa, May 1983.

[3] Fried, W.R., "Principles and Simulation of JTIDS Relative Navigation", *IEEE Transactions on Aerospace and Electronic Systems*, Vol. AES-14, No.1, January 1978.

[4] Tanenbaum, A.S., *Computer Networks*:, Prentice-Hall, 1981.

[5] Baker, D.J., and Ephremides, A., "The Architectural Organization of a Mobile Radio Network via a Distributed Algorithm", *IEEE Trans. Comm.*, Vol. COM-29, No.11, November 1981.

[6] Baker, D.J., "Distributed Control of Broadcast Radio Networks with Changing Topologies", *IEEE Infocom*, April 1983.

[7] Mannel. W.M., "Future Communications Concepts in Support of US Army Command and Control", *IEEE Trans. Comm.*, Vol. COM-28, No.9, September 1980.

[8] Brick, D.B., and Ellersick, F.W., "Future Air Force Tactical Communications", *IEEE Trans. Comm.*, Vol. COM-28, No.9, September 1980.

[9] Shindo, S.Y., Nakamura, Y., and Ogawa, H., "TDMA for Radio Local Distribution System", *International Comm. Conf.*, 1983.

[10] Even, S., Goldriech, O., and Tong, P. "On the NP Completeness of Certain Network Testing Problems", *Technical*Report december 1981. Technion- Israel Institute of Technology, Haifa, Israel. To appear in *Networks*.