To understand why $\tilde{\phi}_d(u)$ is such a good approximation of $\tilde{\phi}(u)$, we rewrite (44) in the independent case as

$$\Pr[\Gamma_{ml} = q, \Gamma_{nl} = p - q] = B(q; J, \mu)B(p - q; J, \mu) \quad (49)$$

with $q = 0, 1, \cdots, p$ for $p = 0, 1, \cdots, J$, and $q = p - J, p - J + 1, \cdots, J$ for $p = J + 1, J + 2, \cdots, 2J$. Then expanding (49) and (47) as power series in μ, we obtain the ratio of the leading terms

$$\frac{J!(J-p)!}{(J-q)!(J-p+q)!} = \frac{J}{J-p+q} \times \frac{J-1}{J-p+q+1} \times \cdots \times \frac{J-q+1}{J-p+1} \cong 1 + \frac{q(p-q)}{J}$$

for $J \gg p$. Therefore, for $\mu \ll 1$, the probabilities decrease very fast with p, and for $p \ll J$, (47) and (49) are in very good agreement.

REFERENCES

[1] D. J. Goodman, P. S. Henry, and V. K. Prabhu, "Frequency-hopped multilevel FSK for mobile radio," *Bell Syst. Tech. J.*, vol. 59, Sept. 1980.

[2] G. R. Cooper and R. W. Nettleton, "A spread-spectrum technique for high-capacity mobile communications," *IEEE Trans. Veh. Technol.*, vol. VT-27, pp. 264–275, Nov. 1978.

[3] O. Yue, "Performance of frequency-hopping multiple-access PSK systems in a Rayleigh fading environment," *Bell Syst. Tech. J.*, vol. 59, July 1980.

[4] R. V. Churchill, *Complex Variables and Applications*, 2nd ed. New York: McGraw-Hill, 1960, pp. 155–159.

[5] O. Yue, "Saddle point approximation for the error probability in PAM systems with intersymbol interference," *IEEE Trans. Commun.*, vol. COM-27, pp. 1604–1609, Oct. 1979.

[6] P. M. Hahn, "Theoretical diversity improvement in multiple frequency shift keying," *IRE Trans. Commun. Syst.*, vol. CS-10, pp. 177–184, June 1962.

[7] D. C. Cox, "910 MHz urban mobile radio propagation: Multipath characteristics in New York City," *IEEE Trans. Commun.*, vol. COM-21, pp. 1188–1194, Nov. 1973.

[8] A. J. Viterbi, *Principles of Coherent Communication*. New York: McGraw-Hill, 1969, ch. 8.

[9] O. Yue, "Hard-limited versus linear combining for frequency-hopping multiple-access systems in a Rayleigh fading environment," *IEEE Trans. Veh. Technol.*, vol. VT-30, pp. 10–14, Mar. 1981.

The Architectural Organization of a Mobile Radio Network via a Distributed Algorithm

DENNIS J. BAKER AND ANTHONY EPHREMIDES, SENIOR MEMBER, IEEE

Abstract—**In this paper we consider the problem of organizing a set of mobile, radio-equipped nodes into a connected network. We require that a reliable structure be acquired and maintained in the face of arbitrary topological changes due to node motion and/or failure. We also require that such a structure be achieved without the use of a central controller. We propose and develop a self-starting, distributed algorithm that establishes and maintains such a connected architecture. This algorithm is especially suited to the needs of the HF Intra-Task Force (ITF) communication network, which is discussed in the paper.**

Paper approved by the Editor for Computer Communication of the IEEE Communications Society for publication after presentation in part at the Conferences on Distributed Computing at INRIA, April 1981, and the Johns Hopkins Conference on Information Sciences and Systems, Johns Hopkins University, Baltimore, MD, March 1981. Manuscript received November 26, 1980; revised May 20, 1981.

D. J. Baker is with the Naval Research Laboratory, Washington, DC 20375.

A. Ephremides is with the Department of Electrical Engineering, University of Maryland, College Park, MD 20742, and the Naval Research Laboratory, Washington, DC 20375.

I. INTRODUCTION

Often a set of mobile, radio-equipped nodes must be organized into a communication network. Examples of such nets include the Packet Radio (PRNET) [7], Advanced Mobile Phone Service (AMPS) [4], Battlefield Information Distribution (BID) [9], and Ptarmigan [8] networks. Each of these networks has many features in common with the others; however, because each also has unique requirements and constraints, their architectural organizations are different. For example, the AMPS network is organized into cellular tessellations with a fixed local controller in each cell. On the other hand, military networks such as the BID and Ptarmigan nets require a more adaptive structure. In the past, the PRNET has used a single station to control the network with several fixed relays used to increase the communication coverage area. Experiments with multistation PRNET's are also underway. A new addition to the list of mobile radio networks is the HF Intra-Task Force (ITF) network [11], which motivated the work described here.

The HF ITF network, now under development, is intended to be a general purpose network providing extended line of sight (ELOS: 50–1000 km) communications for Naval task force units. The nodes in this network will be linked via radio waves from the HF band (2–30 MHz). At HF the radio communication range is not constant but varies in time and frequency due to a variety or factors such as 1) relative node orientation (antenna pattern effects), 2) noise level, 3) propagation losses (sea state effects [2]), and 4) fading (ground wave/sky wave interference). As a result of these variations, the connectivities in the network are subject to change. Furthermore, these connectivities are affected by the relative movement of the nodes, node and link failures, and the addition of new nodes. Consequently, a very important characteristic of a mobile and widely dispersed communication network, such as the ITF network, is its *changing topology*. Therefore, any organization of the network's architecture must account for these changes.

The unique constraints and requirements imposed on the HF ITF net [11] require the development of new techniques for organizing the network. Network survivability considerations favor schemes that allow the network to organize itself into a reliable network structure and then maintain this structure under changing connectivities. Furthermore, this self-organizing capability should not require the existence of a central controller. The peculiarities of the HF channel also prevent any simple adaptation of network architectures designed for implementation in other frequency bands.

A basic characteristic of a radio channel is its broadcast nature. Coupled with the multiple access aspect of the channel, this represents another source of serious complications; namely, transmissions from different nodes may interfere (collide) with each other. If all nodes are in direct contact with a central node, there are several protocols of access that can handle the resolution of these conflicts successfully. However,

0090-6778/81/1100-1694$00.75

if the nodes are widely dispersed, the effect of the "hidden terminal" phenomenon [10] complicates the resolution of transmission conflicts. Consequently, we may identify the need for *conflict resolution capability* as another basic requirement of a mobile and widely spread radio network, such as the ITF network. Again, the network's architectural organization must provide this capability.

The preceding discussion suggests an architecture as shown in Fig. 1. The network is organized into a set of *node clusters*, each node belonging to at least one cluster. Every cluster has its own *cluster head* which acts as a *local controller* for the nodes in that cluster. Since all nodes belonging to a cluster are in direct communication range from the cluster controller, the hidden terminal problem can be solved by the use of existing techniques [10]. In case of nonrandom access techniques within a cluster (such as polling or centralized reservations), the local controller structure permits their use also. The cluster heads are linked via gateways (if needed) to provide paths for intercluster communication and global network connectivity. Since the nodes are in constant motion, we want to develop an algorithm that can *establish* (from any initial node configuration) and *maintain* (for any node motion and/or failure patterns) such an architectural profile as depicted in Fig. 1. Clearly, the algorithm should be expected to achieve this connected organizational structure so long as the nodes have not moved so far apart that the network has become disconnected. Moreover, to ensure survivability of the network, the algorithm must not depend on the existence of any particular node. Thus, the algorithm must be distributed. One must distinguish the network's dependence on the resulting architecture from that of the algorithm. The algorithm simply constitutes the process by which the role of each node is selected. The purpose of this paper is to describe such an algorithm.

The rest of the paper is organized as follows. In Section II we describe the operation of the proposed algorithm, hereafter called the *linked cluster algorithm*. Section III contains simulation examples of networks organized by this algorithm and the different possibilities that may arise. Some secondary issues relating to the implementation of the basic algorithm are discussed in Section IV. The Appendix contains a concise pseudolisting of the linked cluster algorithm.

II. THE LINKED CLUSTER ALGORITHM

The network architecture shown in Fig. 1 is not, by itself, adequate for the HF ITF network. The structure shown in Fig. 1 is based on a single connectivity map for the network. However, over the entire HF band, there may be several different connectivity maps due to variations in the HF communication range with frequency. Consequently, we have considered a network architecture that consists of several overlayed sets of linked clusters, each set being similar to the one shown in Fig. 1 and being based on a particular connectivity map. Moreover, these connectivity maps are continuously being reformed in order to adapt to the time variation of the HF ITF network connectivities. The HF band is partitioned for this reason into M subbands, for each of which a separate run of the algorithm is required in order to produce the corresponding sets of clusters. These separate runs take place consecutively during M epochs. During epoch i the algorithm is run for the ith subband of the HF channel. A connectivity map is formed based on the connectivities that exist within that subband. The algorithm then provides for the selection of cluster heads and gateway nodes. When the M runs are completed, the epochs repeat in a cyclic fashion providing a continual updating process. Note that during any epoch only one set of linked clusters is being reorganized—the remaining $M-1$ sets are unaffected. To prevent disruptions in communication traffic flow, the network should route traffic so as to avoid the subband in which the network is being reorganized. Appropriate message framing provisions must be made, of course, in order to avoid interruption of message transmissions at the beginning of the corresponding reorganization epochs. In the remainder of the paper, when we talk about the algorithm we shall be referring to those steps of the algorithm that occur during any individual epoch.

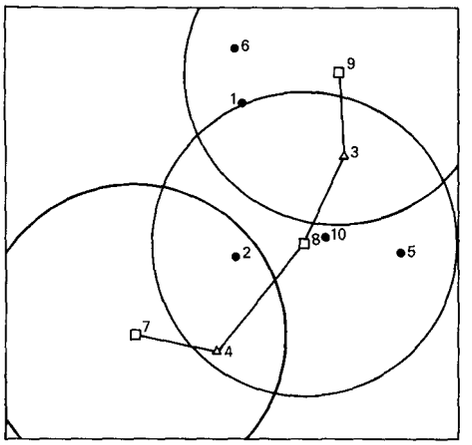

Fig. 1. Example showing proposed organization of a mobile radio network for one epoch. As shown, a node may be either an ordinary node (solid dot), a cluster head (square), or a gateway node (triangle). Cluster heads act as local central controllers. The example is for the case of a fixed communication range, as indicated by the circle around each cluster head. Clusters are linked together by gateways, if necessary, to form a backbone network for intercluster communication.

The algorithm that produces the linked cluster organization has two logical stages: first, the formation of clusters and, second, the linking of the clusters. Each node performs the steps of the algorithm based on *local* information. Thus the algorithm is distributed. Some exchanges of messages are required in this algorithm. These are assumed to be transmitted over a separate channel, called the *control channel*. As depicted in Fig. 1, at the completion of the algorithm the network is organized into linked clusters and each node ends up assuming one of three roles; it may remain an *ordinary* node, it may become a *gateway* node, or it may assume the *cluster head* node position. The significance of these roles will become apparent as the algorithm is described.

The set of links that interconnect the cluster heads forms the so-called *backbone network*. It is not clear whether all or only part of the intercluster traffic should be routed through this backbone network. For example, direct links could be assigned between some nodes belonging to different clusters. This would depend on routing, traffic volume, and delay considerations that are not addressed in this paper. Depending on the intended use of the backbone network, it may be desirable that it be as close to a minimal spanning tree as possible for the purpose of reducing the number of local controllers and relay steps in transmissions, or that it have loop paths and superfluous links for reliability and for the purpose of deconcentrating the intercluster traffic. How the local controllers should coordinate their management of the network resources for routing, flow control, and the like is not of concern to us here. We shall only show how some redundancies can be

removed, if desired, by direct use of our algorithm. Additional reduction of links can be achieved by separate use of other algorithms similar to the ones described by Dalal [3] and Gallager [6].

The schedule of events in the algorithm is shown in Fig. 2. All nodes are assumed to keep accurate global time. Each epoch is divided into two *frames,* each of which is subdivided into N time division multiple access (TDMA) *slots.* Each node is identified by a unique integer from 1 to N and is allowed to transmit control channel messages related to the algorithm only in the correspondingly numbered slot within each frame.

During execution of the linked cluster algorithm each node maintains the following data structures: three lists called, respectively, HEADSONEHOPAWAY, HEADSTWOHOPSAWAY, and NODESHEARD, a matrix called CONNECTIVITY, a variable named OWNHEAD, and an indicator named NODESTATUS. These data structures are updated routinely as control messages are received from other nodes. We first describe what these data structures represent.

The CONNECTIVITY matrix has binary entries. A value of 1 in the (i, j) position indicates the existence of a link between nodes i and j, and a value of 0 the lack of one. Thus, the ith row of the matrix indicates the connectivities of the ith node. It is possible that at the location of node i several rows of the matrix remain unfilled due to the lack of knowledge about the connectivities at distant locations in the network.

The variable OWNHEAD designates the identity of the node that is assigned to be the cluster head for a given node. The NODESTATUS indicator takes on one of three values, ORDINARY, GATEWAY, or HEAD to specify the role assumed by a given node.

The NODESHEARD list includes all neighboring nodes that a node can hear. In some cases, for these neighbors, communication in the reverse direction may not be possible. A built-in feature of our algorithm is the capability to distinguish between unidirectional and bidirectional connectivities.

The HEADSONEHOPAWAY list records those cluster head nodes that are connected to a node and, finally, the HEADSTWOHOPSAWAY list identifies those cluster heads that are not directly connected to a node but which are connected to one of its neighbors.

On a frame by frame basis, the algorithm proceeds as follows.

Frame 1: In its assigned slot (ith), node i *broadcasts* the identities of the nodes it has heard from during the earlier slots of this frame (i.e., its NODESHEARD list). Thus, node i also *receives* partial connectivity information of the nodes that it can hear. So at the end of this frame, node i has filled in some of the entries of its connectivity matrix. In particular, it can fill in elements (i, j) of the ith row that satisfy $j > i$. The element (i, j) is set equal to 1 if node i heard from node j and node i appears in the NODESHEARD list broadcast by node j.

Frame 2: Now each node *broadcasts* in its assigned slot its full connectivity row. This is possible because node i has completely filled in the ith row of the connectivity matrix by the time of the ith slot of Frame 2. Here is how node i determines the bidirectionality of links (i, j) for $j < i$. Node i sets connectivity matrix element (i, j), for $j < i$, equal to 1 if the ith element of the connectivity row received from node j during Frame 2 is equal to 1. So at the end of the frame each

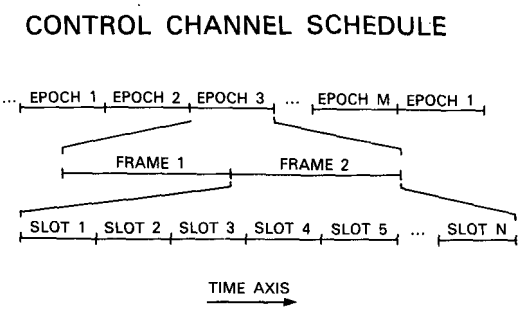

Fig. 2. Schedule for the transmission of control messages. Each node is assigned one of N transmission slots. There are two frames—corresponding to the two types of control messages. The linked cluster algorithm is repeated each epoch. During an epoch, all nodes transmit on the same frequency channel; there are M such channels.

node knows the two-way connectivities for itself and for its immediate neighbors. In this manner, all bidirectional links are determined. From these bidirectional links evolves the linked cluster organization. The global connectivity matrix is not available to every individual node—only a partial version of it is formed by each node. However, if we assume error-free transmissions, all versions are *consistent* with the global matrix.

At the ith slot of the second frame, node i can perform a logical function that permits it to transmit, together with its row connectivity information, its NODESTATUS. We use the rule that the node with the highest identity number among a group of nodes is the first candidate to claim cluster head status. Thus, node i first checks its connectivity row. If there is no neighbor with higher identity number, node i becomes a cluster head. If one or more neighbors exist with higher identity number, the highest numbered neighbor will become a cluster head, so i does not have to. However, i must also check whether it is the "highest" neighbor of some other node $j < i$. This can be done by checking the received connectivity rows from the lower numbered neighbors. If node i is the highest in some row $j < i$, node i must become a cluster head for at least node j. Thus, node i broadcasts in the ith slot of the second frame its current NODESTATUS. This information is needed for the linking of the clusters, as will be seen later. Note that if the selection rule were changed from the choice of the "highest" to that of the "lowest" numbered node as the cluster head, then it would not be possible for a node to know for certain, prior to its Frame 2 transmission, whether it should become a cluster head. Consequently, an additional frame would be needed for the exchange of NODESTATUS information. (See, for example, an earlier version of the linked cluster algorithm [1].)

At the end of the second frame each node fills in its HEADSONEHOPAWAY and HEADSTWOHOPSAWAY lists. The first list consists of those neighbors of the node that achieved cluster head NODESTATUS. For the second list, the node "examines" each of its neighbors. If the cluster head of a neighboring node is not within one hop of this "examining" node, this cluster head is put into the HEADSTWOHOPSAWAY list.

It is worth noting that the procedure HEADNUMBER, which is used for determining the cluster head for a given node, is simply a *distributed* implementation of the following very simple centralized procedure of cluster head selection. Start with the highest numbered node, say node N, and declare it a cluster head. Draw a circle around that node N with radius equal to the range of communication. Are there any nodes left

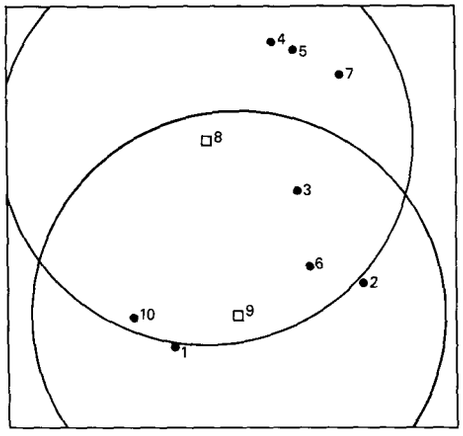

Fig. 3. In this example, cluster heads 8 and 9 can be linked directly.

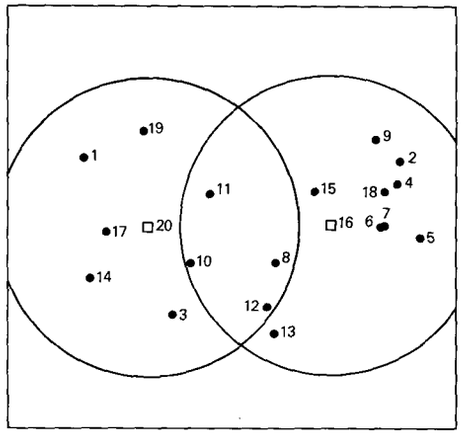

Fig. 4. Example of overlapping clusters. Nodes 8, 10, 11, and 12 are candidates for becoming gateway nodes to link cluster heads 16 and 20.

outside this circle?. If yes, then *tentatively* consider drawing a circle about $N-1$. Should any nodes lie within this circle that were not already within the first circle, node $N-1$ becomes a cluster head and a circle is drawn about it. Otherwise we proceed to consider node $N-2$ by repeating the same procedure, then on to node $N-3$, etc., until all nodes lie within at least one circle. In the end every node has at least one cluster head in its vicinity and there is only one head if all nodes are within a distance of one hop from each other.

Thus, the clusters are formed by the end of the second frame and the database necessary for the second logical function of the algorithm (the linking of the clusters) is also available.

An intermediate procedure, called DELETEHEADS, can be used, before the linking up of the clusters, in order to remove some superfluous cluster heads. The HEADNUMBER procedure may produce a few unnecessary cluster heads under some circumstances. It is not clear whether a loop-free, parsimoniously sparse backbone network is desired or whether a redundant one with plentiful alternative paths is preferred. The resulting set of cluster heads depends on the topological initial condition of the network and can range from one extreme to the other. Procedure DELETEHEADS provides for the elimination of some redundant clusters that may have been formed. Let us say that one cluster *covers* another if all the nodes of the second are also members of the first. Procedure HEADNUMBER will never produce clusters mutually covering each other. However, it is possible for one cluster to cover another. In that case, DELETEHEADS takes over and eliminates the covered cluster head. Each node in the clusters involved can detect this coverage and may proceed to adjust its data base by deleting the head of the redundant cluster from the HEADSONEHOPAWAY list (if the redundant head is a neighbor) or by returning to ORDINARY NODESTATUS (if a head discovers itself to be covered). When a node finds that its own cluster head is covered, the covering node becomes its new OWNHEAD.

The next logical stage is the linking up of the clusters. This is accomplished by the introduction of GATEWAY status to some nodes. Every nonhead node is a candidate to become a GATEWAY. There are three cases of relative positions among clusters. The first one is indicated in Fig. 3. Here there is no need for gateways since the heads of the clusters are directly linked. The second one is depicted in Fig. 4. Here exactly one

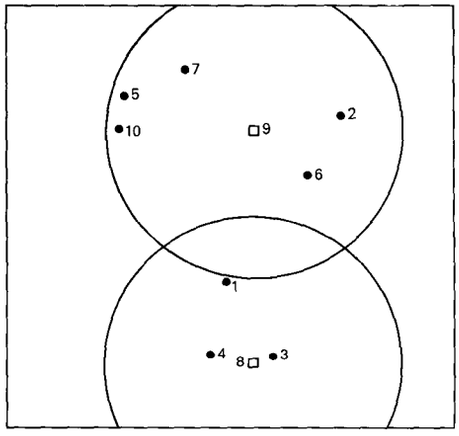

Fig. 5. Example of nonoverlapping clusters. In this case, there are no nodes that are within range of both cluster heads. A gateway node pair must be formed to link the cluster heads—one node from each cluster.

node is needed to link up the two heads. Clearly, the candidates are the nodes in the intersection of the two cluster regions. The third case is pictured in Fig. 5. Here two nodes (one from each cluster) are needed. It is assumed, of course, that suitable such nodes exist; otherwise, the network cannot be connected. In the sequel we describe the procedures used to achieve the linkup in the last two cases.

Case of Overlapping Clusters—This case is handled by procedure LINKUP1. Each node tests all pairs of heads in its HEADSONEHOPAWAY list for connectivity or linkage through a third cluster head. This is easily accomplished by comparison of their connectivity rows. If an unlinked pair of heads is found, the examining node is a candidate gateway for linking these heads. The highest numbered node in the intersection of the two clusters is chosen to become a gateway for that pair. All nodes in this intersection are aware of each other since they can be at most two hops away from each other. Thus, there is no ambiguity in the selection of the gateway node.

Case of Nonoverlapping Clusters—This case is handled by procedure LINKUP2. For linking up two clusters that do not overlap, at least one node from each cluster must become a gateway. Each node examines every possible pair of nodes, the first member of which is its own cluster head and the

second member of which is a cluster head in its HEADSTWOHOPSAWAY list. To avoid creation of redundant gateways, the node attempts to ascertain the need for the creation of a gateway by checking, for each such pair, whether a path may have been already created via LINKUP1 through another cluster head (see Fig. 6). Such a linkage would have occurred if there existed a node among the candidate gateway's neighbors that included in its connectivity row the second member of the head-pair being examined, together with a cluster head from the candidate gateway's HEADSONEHOPAWAY list. If no such circumstance is established, however, the node proceeds further with the LINKUP2 procedure. There may be several pairs of potential gateway nodes that can link two clusters. Each node may be aware of several of those but perhaps not all of them. The (arbitrary) deterministic rule chosen for resolving the ambiguity is to select the pair with the largest sum of identity numbers. In case of a tie, the pair involving the node with the highest number is chosen. Unlike the case of LINKUP1, here we may end up with two or perhaps more gateway pairs. It is worth noting, however, that such multiple linkage outcomes are not very likely for most topological configurations. In some cases, illustrated in the next section, only one of the two potential gateway nodes in a pair may decide to become a gateway, while the other may find that it is not needed if another pair of higher numbered nodes is available. The existence of such a pair may not be known to both partners of the first pair, and thus asymmetric situations can arise. Such outcomes, are rare, however, and constitute only a harmless nuisance. They do not affect the network's operation and cannot be avoided without increasing the databases available at each node by additional message exchanges.

III. SIMULATION EXAMPLES

A simulator model was constructed to provide examples of networks that use the linked cluster algorithm just described, and to depict the resulting organization. In this section we show selected samples of simulation outputs that illustrate several of the possibilities that may arise and the corresponding action by the algorithm. The input parameters for the simulator are the number of nodes, the communication range, a seed number for generating random node positions, and the radius of the circle within which all nodes must lie, that is, the maximum spatial size of the network. The use of a single and constant number for the communication range implies that, for the simulator model, all links are bidirectional and there are no fluctuations due to the causes discussed in the Introduction section. Of course, as explained in Section II, the algorithm itself does not require a bidirectionality assumption.

Fig. 7 shows the connectivities for a network example of 10 nodes. After completion of the second frame of information exchange messages, the network has achieved the organization shown in Fig. 8. Initially, nodes 10, 9, and 8 achieved HEAD status, but procedure DELETEHEADS eliminated node 10 because its cluster was fully covered by that of node 9. Thus, 9 and 8 are the only two cluster heads. The corresponding clusters overlap and are connected via node 3 (the only node in their intersection in this case) which became a GATEWAY. All other nodes remain ORDINARY. Note that node 7 lies dangerously close to the edge of the communication range. If it proceeds in an outward direction relative to node 8, the connectivity will be lost. However, at the next repetition of the algorithm (when the two-frame epoch at the frequency corresponding to this communication range recurs) the structure will adapt to the change, will provde the necessary connectivities, and will maintain a similar profile of overlapping clusters with, perhaps, different cluster heads and gateways.

Fig. 9 shows the resulting organization for an example of

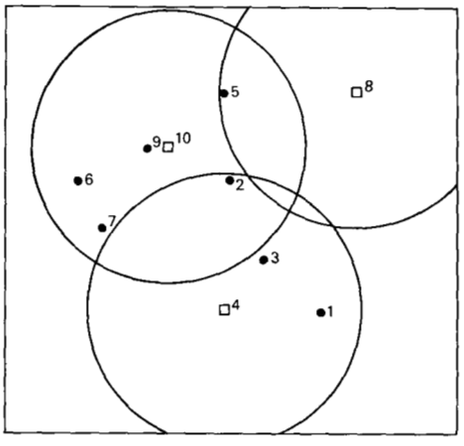

Fig. 6. This illustrates the special case of a cluster (10) overlapping two clusters (4 and 8) that are nonoverlapping. In this case, the nonoverlapping clusters (4 and 8) need not be linked directly via the formation of a gateway node pair since cluster heads 4 and 8 can be linked via the overlapping cluster (10).

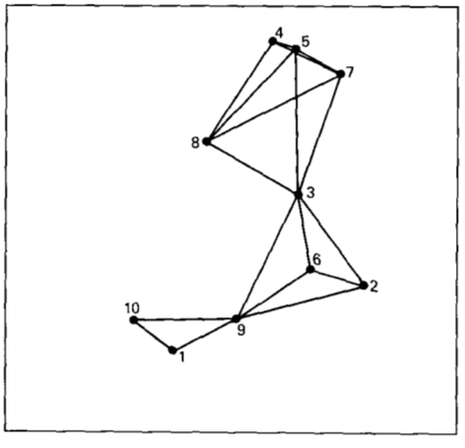

Fig. 7. Connectivities for a sample, 10 node, network.

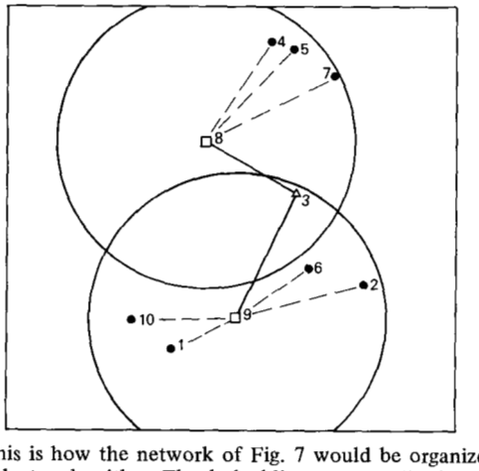

Fig. 8. This is how the network of Fig. 7 would be organized by the linked cluster algorithm. The dashed lines connect "ordinary" nodes to their respective cluster heads.

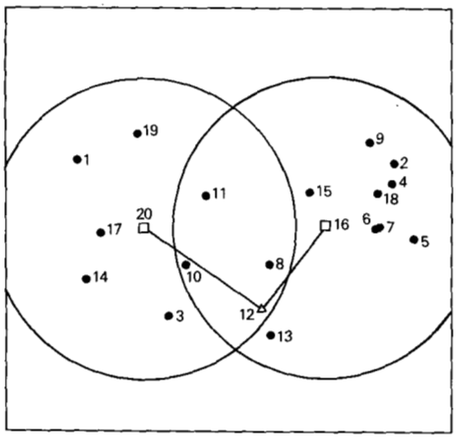

Fig. 9. Example showing how gateway is selected for overlapping clusters.

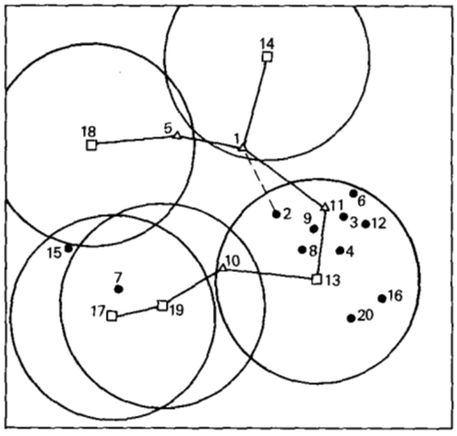

Fig. 10. Example showing the linking of nonoverlapping clusters. Node pair (1, 2), connected by the dashed line, represents a potential, but unrealized, gateway node pair.

20 nodes. Again only two clusters are formed. (Initially three clusters were formed, node 18 being also a cluster head; however, that cluster was completely covered by the cluster of HEAD node 16 and was eliminated by the procedure DELETE-HEADS.)

Figs. 10 and 11 illustrate the case of nonoverlapping clusters. In the case of Fig. 10 there are five cluster heads, nodes 19, 18, 17, 14, and 13 (the bias toward selecting high numbered nodes for cluster heads is, of course, due to the deliberate rule used by the procedure HEADNUMBER). Clusters 17 and 19 connect without need for gateways. Node 10 is a gateway linking clusters 19 and 13. Clusters 17 and 18 cannot be linked and, were it not for cluster 14, the nodes of cluster 18 would be disconnected from the network. As it stands, however, clusters 14 and 18 are linked via gateways 1 and 5, while clusters 13 and 14 are linked via 11 and 1. The dotted line between nodes 1 and 2 shows another potential pair of gateways for the linkage of clusters 13 and 14. However, the sum of their identities (equal to 3) is less than the sum of nodes 1 and 11, and thus they were not selected to serve as gateways. The links shown in Fig. 11 form the backbone network.

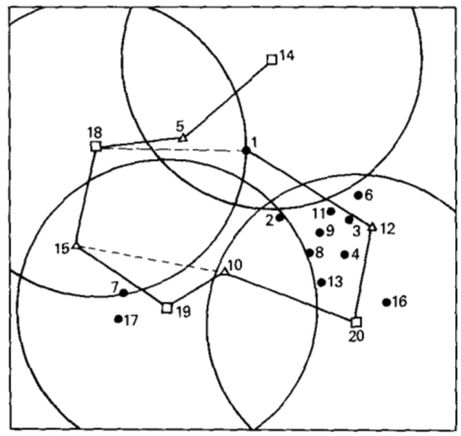

Fig. 11. The linked cluster algorithm eliminates some redundant links on the backbone network, such as link (10, 15) shown here. On the other hand, some asymmetric gateway node pairs, such as (1, 12), can also occur.

In the case of Fig. 11 there are four cluster heads, nodes 20, 19, 18, and 14. Node 15 lies in the intersection of clusters 18 and 19 and is a gateway linking them; so is node 10 for clusters 19 and 20 and node 5 for clusters 18 and 14. Additional interesting developments are exhibited in this figure. Node 10, in its search to link nonoverlapping clusters, formed a potential pair of gateways with node 15 for linking clusters 18 and 20. However, in their search, both nodes established that clusterhead 19 lies in the connectivity row of each one of them and, thus, they would be superfluous gateways. The dotted line connecting 10 and 15 indicates the potential backbone network link that did not materialize.

A slightly different scenario developed in the case of nodes 1 and 12 in their search for linking the same clusters 18 and 20. Node 1 is aware of nodes 10 and 15 as a candidate pair of gateways for these clusters, while node 12 is not. Thus, node 1 computes both sums ($1 + 12 = 13$ and $15 + 10 = 25$) and determines that its services as a gateway are unnecessary, while node 12, unaware of the (10, 15) pair, determines that it should form a pair with node 1. Thus, links (12, 1) and (12, 20) are established by 12 as part of the backbone network. Node 1, on the other hand, does not set up the link between 18 and 1 (indicated by the dotted line). This is an instance of the asymmetric development discussed at the end of the preceding section. Since gateways have only a passive relaying function, in contrast to the cluster heads who take an active controller's role, no harm is done by such an occurrence. However, it represents a weakness of the algorithm, a typical one for cases of distributed databases. The elimination of such superfluous, half-way links is desirable, but the matter is not pursued further in this paper. Note that if node 1 was not aware of the (10, 15) pair, we would end up with two pairs of gateways for clusters 18 and 20. Such a situation is not harmful, may arise occasionally, and can be tolerated. Repeated simulations as well as rigorous validation failed to produce any weaknesses or potential malfunctions of the algorithm. Thus, the examples examined here just about cover all interesting cases and show the kinds of organizations that would typically occur.

If the communication range varies significantly across the frequency band allocated to the network, the resulting organization may differ considerably from epoch to epoch. For example, the network may be disconnected at one frequency with relatively short communication range, while it stays connected at another frequency with longer range.

It is not unusual for the linked cluster algorithm to produce backbone networks containing cycles. They are especially

likely to occur when the communication range is comparable to the average separation of the nodes.

IV. FURTHER CONSIDERATIONS

Once the organization described above is achieved via the linked cluster algorithm, there are several issues that arise relative to the operation of the network. In this section we offer a preliminary discussion of some of them.

1) *Intracluster Communication:* The cluster structure is ideally suited to the use of the cluster head as a local controller for the nodes in its cluster. Thus, all messages originating from ordinary nodes must be first transmitted to their respective cluster heads for further handling and routing. Nevertheless, a cluster head could allocate a direct link between two nodes in its cluster, in a circuit-switched fashion, if there was need for prolonged voice conversation or lengthy file transfers between these nodes. Accessing of the cluster head by the ordinary nodes in its region could take place on a random access basis. Since all cluster members can hear the cluster head, protocols such as the busy tone multiple access (BTMA) can be used. Such protocols are capable of solving the "hidden terminal" problem that is so typical in mobile radio telecommunication networks.

2) *Intercluster Communication:* It is natural, under the cluster structure, to assign exclusive routing control to cluster head and gateway nodes only. Therefore, considerable concentration of traffic would occur on the links of the backbone network. For this reason, traffic over the backbone network would be less bursty than intracluster traffic. Thus, random access protocols would not be appropriate. Another factor arguing against the use of random access schemes for intercluster traffic is the hidden terminal problem. The busy tone solution is not applicable at this communication level. It is also remarkable that the resulting concentration of traffic is not contributing as much to an unbalanced use of the network resources as it might appear at first sight. If the average node separation is large, there are generally many clusters—each with few members in it. Thus, the occurring concentration is primarily due to multihop routing and not so much due to direct concentration of the member nodes' messages. Assuming equitable distribution of traffic requirements for each node pair, this kind of routing-caused concentration would not be too unbalanced. On the other hand, if the average node separation is small, there are usually few clusters, each with several nodes under its jurisdiction. In this case it may appear that the resulting concentration is considerable. However, the cluster heads can easily assign several direct links between communication nodes that are within close range of each other, thus relieving the backbone network links from a disproportionate share of the traffic load.

3) *Node Numbering:* Given the simple strategy used in deciding the identity of a cluster head and/or a gateway node among a group of candidates, it is clear that the method of number assignment to the nodes is a very important part of the proposed organization. Higher numbered nodes simply have a greater tendency to become heads or acquire gateway status than lower numbered nodes. In particular, the highest numbered node will almost always become a cluster head. The only time it will not is when it is covered completely by another cluster head. If the numbering system is the same for all frequencies, these nodes will be overburdened with network management and traffic direction responsibilities. One possible way to alleviate the problem is to assign to each node a different number for each frequency. Also, if all nodes do not have comparable equipment, the higher numbers should be assigned to nodes with greater capability.

4) *Addition and Deletion of Nodes:* The linked cluster algorithm depends directly on the orderly exchange of messages during the two frames of every epoch. These transmissions take place on a TDMA basis. Thus, the most fundamental assumptions of the proposed algorithm are that each node keeps accurate common time and knows the precise length (in number of slots) of each frame. The first assumption is not questioned at this point. Guard times can be provided to accommodate small fluctuations in the clocks under plesiochronous operation. The second assumption, however, depends critically on the number of nodes in the network. If the maximum possible number is known *a-priori,* enough slots may be provided in each frame for the worst case when all possible users are present in the network. This, of course, is a highly inefficient strategy and may not even be applicable if the number of users cannot be bounded with certainty. A modification or an extension of the algorithm is necessary in order to permit the occasional adjustment of the frame length. Such an adjustment, whenever it occurs, constitutes a drastic alteration of a very critical parameter of the algorithm. Deadlocks and utter confusion may result if *all* nodes do not make the *same* adjustment at the *same* time. This problem is not addressed here, as its complexity extends beyond the scope of this paper.

5) *Errors and Their Effects:* It is convenient to present first the linked cluster algorithm in the context of an error-free environment, as we have done here. However, although several techniques exist for overcoming some channel errors, any algorithm that relies on message exchanges between nodes via radio waves must be able to withstand communication errors. In the linked cluster algorithm channel errors will result in the incorrect assignment of cluster head and gateway status to some nodes. It is quite straightforward to investigate these structural aberrations via simulation techniques, and this we have done. However, the importance of these aberrations cannot be evaluated without also modeling other features of the network—especially routing. On the other hand, by investigating the effects of errors on the resulting network structures, we hope to be able to design a better routing strategy.

V. CONCLUSION

The variable topological connectivity in a mobile radio telecommunication network necessitates the invention of suitable organizational structures for the network and preferably of distributed algorithms to achieve these structures. In this paper we have introduced an architecture of linked clusters which, via the proposed linked cluster algorithm, adapts to the changes and enables the network to remodel itself and maintain balanced connectivities. Simulations that illustrate the different cases that may arise have been conducted, and selected results have been included in this paper. Finally, we have presented a preliminary discussion of issues related to the overall network operation after the acquisition of the proposed architectural profile.

The linked cluster algorithm was developed for use in the HF ITF network. Features of this algorithm that make it particularly attractive for application to the ITF net include the following: 1) the algorithm requires no central controller and is fully distributed, 2) network connectivities are periodically monitored and a reliable network structure is formed based on existing links and nodes, 3) several connectivity maps are formed corresponding to various subbands of the HF band,

4) the resulting network structures can be used to implement a wide variety of routing strategies, and 5) the cluster structure can be used to avoid the problem of "hidden terminals." An earlier version of this algorithm can be found in [1], and a slightly modified version was presented in [5]. Other algorithms for mobile radio networks can be found in [7] and [8]; however, these are not applicable in any straightforward way to the HF ITF network, as was explained in the Introduction.

APPENDIX

In this Appendix we present a concise pseudolisting of the linked cluster algorithm. Every node executes this algorithm once during each epoch.

```
BEGIN

COMMENT: Initialize data structures;
OWNHEAD: =SELF;
NODESTATUS: =HEAD;
HEADSONEHOPAWAY: =EMPTY;
HEADSTWOHOPSAWAY: =EMPTY;
NODESHEARD: =EMPTY;
CONNECTIVITY: =IDENTITYMATRIX;

Schedule transmit event processes at this node, call it node i;

WHILE not end of Frame 2 wait for one of the following
events to occur and then process as indicated below

  BEGIN

    COMMENT: Frame 1 events;

    Transmit:   Broadcast NODESHEARD list;

    Receive:    COMMENT: "Nodes-Heard" list
                received from node j;
                Put node j into NODESHEARD
                list;
                IF node i was heard by node j
                THEN set CONNECTIVITY [i,
                j]: = 1;

    COMMENT: Frame 2 events:

    Transmit:   Determine NODESTATUS;
                Broadcast row i of CONNEC-
                TIVITY matrix and NODE-
                STATUS;

    Receive:    COMMENT: "Connectivity/Sta-
                tus" message received from node
                j;
                Fill in row j of CONNECTIVITY
                matrix;
                IF j<i, set CONNECTIVITY
                [i, j]: =CONNECTIVITY [j,
                i];

                IF status of node j = HEAD;
                THEN put node j into HEADS-
                ONEHOPAWAY;

  END;

COMMENT: Fill in HEADSTWOHOPSAWAY list;

For each node m that is bidirectionally linked to node i DO
  BEGIN
      Find HEAD for node m—call it node k;
      IF k NE i and CONNECTIVITY [i, k] = 0
      THEN put node k into HEADSTWOHOPS-
      AWAY;
  END;

COMMENT: Call procedures to perform cluster linkage;
DELETEHEADS;
LINKUP1;
LINKUP2;

END;
```

REFERENCES

[1] D. J. Baker and A. Ephremides, "A distributed algorithm for organizing mobile radio telecommunication networks," in *Proc. 2nd Int. Conf. Distributed Comput. Syst.*, Paris, France, Apr. 8–10, 1981.

[2] D. Barrick, "Theory of HF and VHF propagation across the rough sea, 2, Application to HF and VHF propagation above the sea," *Radio Sci.*, vol. 6, pp. 527–533, May 1971.

[3] Y. K. Dalal, "Broadcast protocols in packet switched computer networks," Dep. Elec. Eng., Stanford Univ., Stanford, CA, Tech. Rep. 128, Apr. 1977.

[4] N. Ehrlich, "The Advanced Mobile Phone Service," *IEEE Commun. Magazine*, vol. 17, pp. 9–15, Mar. 1979.

[5] A. Ephremides and D. Baker, "An alternative algorithm for the distributed organization of mobile users into connected networks," presented at the Conf. Inform. Sci. Syst., Johns Hopkins Univ., Baltimore, MD, Mar. 1981.

[6] R. G. Gallager, P. A. Humblet, and P. M. Spira, "A distributed algorithm for minimum weight spanning trees," Lab. Inform. Decision Syst., Massachusetts Inst. of Technol., Cambridge, Tech. Rep. LIDS-P-906-A, Oct. 1979.

[7] R. E. Kahn, S. A. Gronemeyer, J. Burchfiel, and R. C. Kunzelman, "Advances in packet radio technology," *Proc. IEEE*, vol. 66, pp. 1468–1496, Nov. 1978.

[8] M. Lawson and S. Smith, "Reconfiguration techniques of a mobile network," in *Proc. 1980 Int. Zurich Sem. Digital Commun.*, Zurich, Switzerland, IEEE 80CH1521-4 COM, 1980, pp. B10.1–B10.5.

[9] A. Nilsson, W. Chon, and C. J. Graff, "A packet radio communication system architecture in a mixed traffic and dynamic environment," in *Proc. Comput. Networking Symp.*, IEEE CH1586-7/80, 1980, pp. 51–66.

[10] F. A. Tobagi and L. Kleinrock, "Packet switching in radio channels: Part II—The hidden terminal problem in carrier sense multiple access and the busy tone solution," *IEEE Trans. Commun.*, vol. COM-23, pp. 1417–1433, Dec. 1975.

[11] J. E. Wieselthier, D. J. Baker, and A. Ephremides, "Survey of problems in the design of an HF Intra-Task Force communication network," Naval Res. Lab. (NRL), Washington, DC, Rep. 8501, 1981.

A Broadcast Protocol for File Transfers to Multiple Sites

S. B. CALO, MEMBER, IEEE, AND M. C. EASTON

***Abstract*—A retransmission protocol for a broadcast connection (point-to-multipoint) is proposed and its performance characteristics are considered. The protocol is designed for transfers of large files**

Paper approved by the Editor for Computer Communication of the IEEE Communications Society for publication without oral presentation. Manuscript received March 8, 1981; revised June 1, 1981.

S. B. Calo is with the IBM Research Center, Yorktown Heights, NY 10598.

M. C. Easton was with the IBM Research Center, Yorktown Heights, NY 10598. He is now with the IBM San Jose Research Lab, San Jose, CA 95193.

0090-6778/81/1100-1701$00.75