NRL Report 8638

An Architecture for the High-Frequency Intratask Force (ITF) Communication Network

D. J. BAKER, A. EPHREMIDES, AND J. E. WIESELTHIER

Communication Systems Engineering Branch
Information Technology Division

December 22, 1982

APPROVED FOR PUBLIC
RELEASE - DISTRIBUTION
UNLIMITED

NAVAL RESEARCH LABORATORY
Washington, D.C.

Research Reports Laboratory
Technical Library
Research Reports Section

DATE: October 23, 2006

FROM: Judi Griffin, Code 5596.3

TO: Raymond Cole code 5520

CC:

SUBJECT: **Review of NRL FORMAL Reports 8638, 8853, 8637"** **for possible distribution changes**

Dear Sir/Madam

1. Please review the enclosed documents for possible distribution changes

(enclosed copies of FR-8637, FR-8853, FR-8638)

Thank You

Judi Griffin
(202) 767-3425
judig@library.nrl.navy.mil

Approved for unlimited distribution: yes ✓ no _____

Comments:

Signature Ray Cole 1/19/07

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER NRL Report 8638	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) AN ARCHITECTURE FOR THE HIGH FREQUENCY INTRATASK FORCE (ITF) COMMUNICATION NETWORK		5. TYPE OF REPORT & PERIOD COVERED Interim report on a continuing NRL problem 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) D. J. Baker, A. Ephremides, and J. E. Wieselthier		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Research Laboratory Washington, DC 20375		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 75-0139A 21-222-805
11. CONTROLLING OFFICE NAME AND ADDRESS Office of Naval Research Arlington, VA 22217		12. REPORT DATE December 22, 1982 13. NUMBER OF PAGES 42
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) UNCLASSIFIED 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Distribution limited to U.S. Government agencies only; test and evaluation; December 1982. Other requests for this document must be referred to the Commanding Officer, Naval Research Laboratory, Washington, DC 20375.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Communications network	Digital simulation
High frequency (HF)	Algorithms
Task force	Adaptive communications
Military communications	Self-organizing systems

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

An architecture is proposed for the HF Intratask Force (ITF) Communication Network. The architecture is achieved by a distributed algorithm that organizes the task force elements into a reliable, highly survivable, network. The resultant network structure, which we call *linked clusters*, is especially suited to telecommunication networks, such as the ITF network, which are characterized by dynamically changing topologies. Network connectivities are periodically monitored, and the algorithm forms a network based on these connectivities. The architectural profile consists of node

(Continued)

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

20. ABSTRACT (Continued)

clusters that are linked to each other, often through gateway nodes. Within each cluster all nodes are connected to a common node; thus, intracluster communication can be governed by appropriate multiple access and broadcast protocols. Intercluster communication takes place over a backbone network that spans the task force. The use of spread spectrum signaling techniques as well as the equipment constraints introduces additional problems of code assignment, distribution, and communication over the backbone network, which are discussed in this report.

CONTENTS

Executive Summary iv

INTRODUCTION 1

Background on Similar Networks 1
Report Outline 2

NETWORK ORGANIZATION 2

Cluster Head Selection Rule (LCA) — Centralized Version 5
Cluster Head Selection Rule (ALCA) — Centralized Version 5
The Linked Cluster Algorithm (LCA) — Distributed Version 6
The Alternative Linked Cluster Algorithm (ALCA) — Distributed Version 11

THE SIMULATION MODEL AND SIMULATION RESULTS 12

Backbone Network Connectivity 13
Undetected Two-Hops-Away Heads 14
Frequency Dependence 14
Node Numbering 14
Dead-End Links 18
Cyclic Paths in Backbone Network 20
Loss of Nodes 21
Loss of Links 23

FURTHER CONSIDERATIONS 26

Intracluster, Backbone, and Auxiliary Links 26
Routing 27
Spread Spectrum Signaling 28
Switching 31
Role of Aircraft and Submarines 32

CONCLUSIONS 32

REFERENCES 32

APPENDIX — Pseudocode for the Network Organizing Algorithms 34

EXECUTIVE SUMMARY

The High-Frequency (HF) Intratask Force (ITF) Network is being designed as a survivable, anti-jam communication network for interconnecting mobile, unsurvivable task force platforms.

We propose an organizational architecture for the ITF network. Other design issues including routing, switching, flow control, and security can be embedded easily and studied in the framework of this architecture. The desire to provide a framework in which other networking issues could be evaluated motivated our early consideration of a network architecture. To assure network survivability, the proposed architecture is based on the use of a distributed algorithm that enables the task force nodes to self-organize into a reliable network structure. The algorithm also provides for the monitoring of network connectivities and for the maintenance of the resultant network organization.

The proposed architectural profile of the network at any given moment consists of node clusters that are linked to each other, as shown in Fig. 1. Here the network is organized into a set of clusters with each node belonging to at least one cluster. Every node is assigned its own cluster head. The cluster head may act as a local controller for all nodes belonging to its cluster. The cluster heads are linked via gateways (as needed) to provide paths for intercluster communication and overall network connectivity. The cluster heads and gateway nodes and the links that interconnect them form the *backbone network.*

We describe two possible versions of an algorithm for producing the linked cluster architecture. The main difference between our two network structuring algorithms, which we refer to as the linked cluster algorithm (LCA) and the alternative linked cluster algorithm (ALCA), is the rule for selecting cluster heads. Both algorithms are distributed and consist of two parts: a communication part, during which local messages are exchanged between neighboring nodes over a *control* channel, and a computation part, which takes place at the end of the communication stage during which each node locally manipulates the collected data base for the determination of its role in the network. The ALCA was developed after the LCA. The ALCA uses a simpler procedure for selecting cluster heads that gives a more uniform geographical distribution of cluster heads. On the other hand, the LCA generally yields network structures containing fewer cluster heads and fewer gateways.

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 overlaid sets of linked clusters. Each set is similar to the one shown in Fig. 1 and is based on a particular connectivity map. Moreover, these connectivity maps are continually being re-formed in order to adapt to the time variation of the HF 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. During any epoch only one set of linked clusters is being reorganized — the remaining M-1 sets are unaffected.

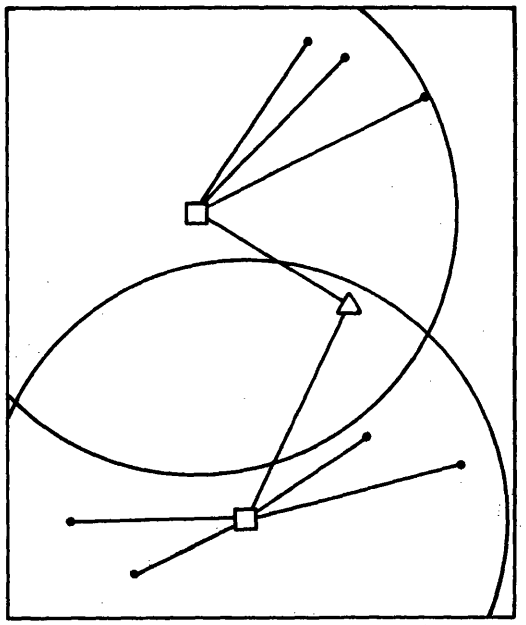

Fig. 1 — Example showing proposed organization of a task force for one epoch. As shown, a node may be either an ordinary node (solid dot), a cluster head (square), or a gateway node (triangle). Cluster heads may act as local central controllers. The example is for the case of a fixed communication range, 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.

Each node performs the steps of the algorithm based on information obtained from neighboring nodes. During each epoch, a time division multiple access (TDMA) scheme governs communication over the *control* channel. Prior to the start of a new epoch, the control channel switches to another frequency subband.

Each epoch is divided into two frames, each of which is subdivided into N TDMA slots. Each node is identified by an 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. Through continual monitoring of possible changes, the network has the capability to adapt its configuration and ensure its survivability. During each epoch regular communication is allowed to take place at all other frequencies except the one corresponding to that epoch. Scheduling the epochs for different times provides the needed robustness as it allows the ITF network to maintain regular communications in most subbands while it adapts to connectivity changes in one subband.

A simulation model was constructed to provide examples of network structures obtained with both algorithms and to illustrate the following features of the linked cluster architecture.

The linked cluster architecture enables the ITF net to self-organize into a connected network despite node and link losses. The technique of overlaying several sets of linked clusters provides alternative communication paths. If a backbone network node or link is lost (e.g. due to jamming), other backbone networks at other frequencies can be used. When the net is reorganized in the subband in which the loss occurs, a new backbone network will be set up that will not contain the disabled node or the jammed link. The combination of path redundancy and the ability of the network to adapt to node and link losses results in a robust ITF net.

Our network structuring algorithms prevent the formation of certain types of loop paths in the backbone network. This reduces the number of links that are needed to maintain the overall connectivity of the network and also simplifies routing.

We show how a node renumbering strategy can be used to produce "orthogonal" backbone networks. This is done to prevent a node from being overburdened with network management and traffic direction responsibilities, which would result from a node always becoming a cluster head or gateway node.

Examples are given indicating that our network structuring algorithms produce efficient network structures even when connectivities are changing during the reorganization epoch. In many cases only a small number of the potentially available links are needed to form a connected network.

The linked cluster architecture is flexible. Our network structuring algorithms provide each node with the following information for each of the M subbands: its one-hop- and two-hops-away neighbors, its NODESTATUS (i.e. whether the node is a cluster head, gateway, or ordinary node), and the identity of its own cluster head. This information can be used in a variety of ways to implement a working network. For example, each node could specify the relay to be used for two-hop paths or the cluster head could make all routing determinations. The flexibility of the linked cluster architecture allows for many different implementations of the ITF network. It is important for the ITF network's architecture to be flexible because some of the equipment constraints that could affect the network design (e.g. synchronization time and the numbers of transmitters and receivers on each platform) are not known precisely.

Some specific networking issues that can be addressed within the framework of the linked cluster architecture include: intra- and intercluster communication, link activation, the use of one-way links, and routing. A preliminary discussion of these issues is given in this report; however, considerable work remains before the design of a complete ITF network is achieved.

Features of the linked cluster architecture that make it ideal for the ITF network are:

- The resultant network structure is survivable since the network structuring algorithms are fully distributed (i.e., there is no need for a central controller).
- The network is self-organizing.
- Network connectivities are continually monitored and the network automatically adapts to node and link losses.
- A variety of multi-access protocols can be used.
- Cluster heads can act as channel access controllers (e.g. a busy tone multiple access (BTMA) scheme can be used which will solve the *hidden terminal* problem).
- Cluster heads can act as broadcast centers.
- Overall network connectivity is achieved using a backbone network.
- The network can be implemented with either narrowband or wideband signaling.
- A variety of routing strategies can be used.
- The architecture takes advantage of the wide variation of the communication range over the HF band.

AN ARCHITECTURE FOR THE HIGH-FREQUENCY INTRATASK FORCE (ITF) COMMUNICATION NETWORK

INTRODUCTION

The trend toward more sophisticated, highly computerized naval platforms is generating new communication requirements that cannot be met by existing communication systems. At the Naval Research Laboratory (NRL) a new communication network is being studied; this new network is a candidate to become the primary extended line of sight (ELOS; 50 to 1000 km) communication system for ships, submarines, and aircraft that form a task force. This new system, called the HF Intratask force (ITF) Communication Network, will use HF radio waves (2 to 30 MHz) to interconnect the various task force platforms.

The HF ITF network can be characterized as a general purpose military communication network with time-varying connectivity that uses broadcast HF radio waves to link nodes. This combination of characteristics makes the HF ITF network unique. There are a few existing networks that share some of these characteristics but none that possess all of them. The HF ITF network must handle both voice and data traffic, which may be either bursty or nonbursty. Some of the communication traffic may require delivery to the destination within a few seconds while other traffic may tolerate delays of hours. Communication traffic classification levels vary from unclassified to highly classified. The network must support several modes of communication including point-to-point, broadcast, and conferencing. Network survivability is an important consideration since the network may be subject to communication jamming, physical attack, spoofing, etc.

Network connectivities vary with frequency and time. One reason for the changing connectivities is node mobility. The task force consists of mobile nodes of widely varying speed capabilities — from relatively slow moving submarines to high-speed aircraft. Network connectivities also vary over the HF band because of the wide variation of HF propagation losses as a function of frequency and sea state [1]. Noise conditions throughout the network may also vary substantially with respect to location, time, sea state, and radio frequency [2]. In addition to atmospheric noise, the network must function in the presence of man-made noise due to jammers, locally generated platform interference, interference from other HF ITF users, and other nonhostile users of the HF medium. Finally, physical attack can remove nodes from the network. A previous report [3] details the requirements, constraints, and networking issues related to the HF ITF network.

Background on Similar Networks

Perhaps the most developed network, which shares many, but not all, of the characteristics of the ITF network, is the packet radio network (PRNET). The version of the PRNET described in Ref. 4 consists of station, relay, and terminal nodes. These nodes are connected by UHF radio links. Initial PRNET experiments were for a single station network where the station acted as a central controller controlling, for example, all network routing. Because it relies on a central controller, this version of the PRNET is not suitable for the ITF network since the loss of the central controller would disable the entire network. Experiments with a multistation PRNET are underway. Unfortunately, the characteristics of the UHF channel are so different from those of the HF channel that generalizations from the PRNET to the HF ITF network are often inappropriate.

Manuscript submitted, July 20, 1982.

The advanced mobile phone service (AMPS) is a commercial radio network that also bears some resemblences to the ITF network. AMPS is organized into cellular tessellations with a fixed local controller in each cell [5]. Although this structure has many operational advantages, it is difficult to maintain when the local controllers are mobile, as would be the case for the ITF network.

The battlefield information distribution (BID) network is another network that shares many of the characteristics of the ITF network. A version of the BID network is described in Ref. 6. The BID network is based on an adaptation of the concepts of the PRNET to a military network environment. It assumes no specific frequency band for communications. It does assume, however, that every node can communicate with a large number of other nodes. Typical node densities are on the order of 500 nodes in an area 30 km by 50 km. In the hierarchical distributed control version of the BID network, a set of nodes is selected as control nodes and each noncontrol node is homed to two or three control nodes. This control structure has some resemblance to the structure that we propose for the ITF network; however, because node densities in the ITF network can be much lower than those assumed for the BID network, and due to the unique properties of the HF channel, the techniques used in the BID network are not directly applicable to our networking problem.

Another mobile, military network is the Ptarmigan communication system. This system is designed to replace the Bruin network currently in use by the British forces assigned to NATO [7]. The Ptarmigan network consists of fixed and mobile nodes and radio and wire communication links. Switching nodes in the Ptarmigan network may move as frequently as once per eight hours. On the other hand, all the nodes in the HF ITF network are moving continuously.

Report Outline

In this report, we propose an organizational architecture for the ITF network. Other design issues including routing, switching, flow control, and security can be easily embedded and studied in the framework of this architecture. The architecture is based on the use of a distributed algorithm, which organizes the task force elements into a reliable network structure. We describe two possible versions of such an algorithm. The proposed structure of the network at any given moment consists of node clusters that are linked to each other. The algorithm provides for the monitoring of network connectivities and for the maintenance of the resultant network organization.

The rest of the report is arranged as follows. The second section describes the two candidate algorithms for organizing the HF ITF Network. These are referred to as the linked cluster algorithm (LCA) and the alternative linked cluster algorithm (ALCA). Examples of networks organized with these algorithms appear in the third section along with a description of our simulation model. Numerous issues related to the architecture are discussed under further considerations followed by our conclusions. The appendix contains concise, SIMULA-based, pseudocode for both algorithms.

NETWORK ORGANIZATION

We consider the problem of constructing a reliable communication network for the example shown in Fig. 2. The resultant network connectivities are shown in Fig. 3 for six different communication ranges corresponding to six subbands of the HF band. The communication ranges shown here are representative groundwave ranges that might apply to task force scenarios in the absence of jamming. Actual connectivities will be different, of course, since the communication range depends on many factors that vary with time and position, including: jamming, sea state, atmospheric noise levels, and other user interference.

As shown in Fig. 3, the ITF network will generally not be completely connected (i.e., not all nodes will be within communication range of every other node), nor will all nodes be within communication range of a central node. The relaying of messages will thus often be necessary. In developing a

network control structure that is well suited to the ITF environment, a fundamental choice is that of distributed versus centralized operation. The main disadvantage of centralized control is its high vulnerability. On the other hand, a significant disadvantage of distributed control is that each platform must receive sufficient information from the other platforms on which to base control decisions; the transfer of all information is subject to delay and possible errors and requires the use of network resources.

Instead of considering a purely centralized or distributed organizational structure, we propose the hybrid architectural organization shown in Fig. 4. Here the network is organized into a set of clusters, each node belonging to at least one cluster. Every node is assigned its own cluster head. The cluster head acts as a local controller for all nodes belonging to its cluster. Since all nodes belonging to a cluster can "hear" the cluster controller, random channel accessing techniques, such as busy tone multiple access (BTMA) [8], designed to solve the *hidden terminal* problem, can be used within the cluster. Similarly centralized polling or other reservation techniques may be used within a cluster. The cluster heads are linked via gateways (if necessary) to provide paths for intercluster communication and overall network connectivity.

In this section we describe two candidate algorithms for establishing and maintaining the linked cluster structure exemplified in Fig. 4. Since connectivities are different for each subband of the HF band, distinct structures will form in each subband. These structures will be subject to changes due to time varying factors including node mobility and jamming. These algorithms are particularly attractive because they do not depend on the existence of any particular node.

The main difference between our two network structuring algorithms, which we refer to as LCA and ALCA, is the method of selecting cluster heads. After the LCA was developed, we found a simpler rule for selecting cluster heads; this new rule, which is used in the ALCA, also has the advantage that it gives a more uniform geographical distribution of cluster heads. On the other hand, the new cluster head selection rule generally yields networks with more heads and gateways than comparable network structures obtained using the LCA. To illustrate the difference between the two selection rules, it is instructive first to describe the two algorithms in a fictitious, centralized mode. That is, we shall temporarily assume that a central controller has full topological and connectivity information about the

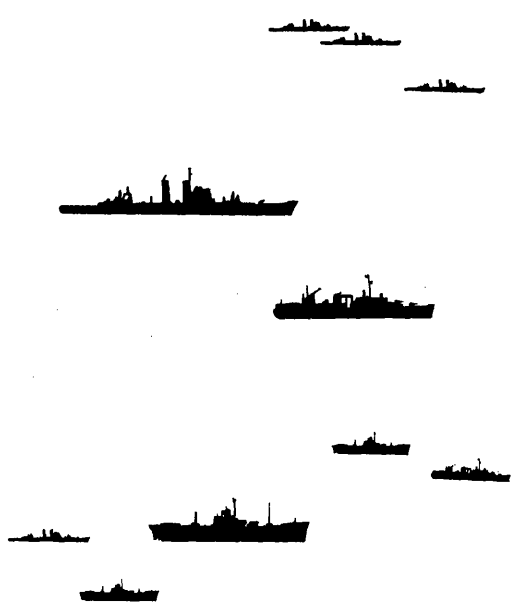

Fig. 2 — We consider the problem of constructing a reliable communication network structure for the example above

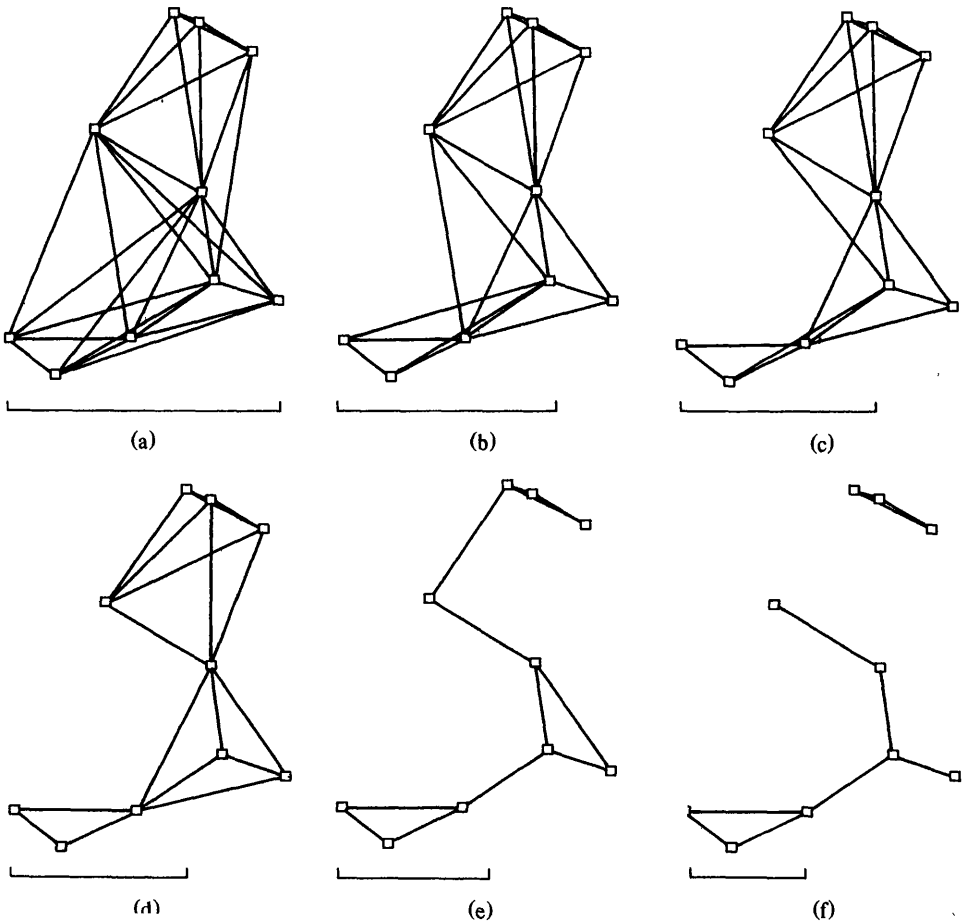

Fig 3 — Connectivity maps for the example of Fig 2 and for six communication ranges The relative magnitude of the communication range is shown below each map

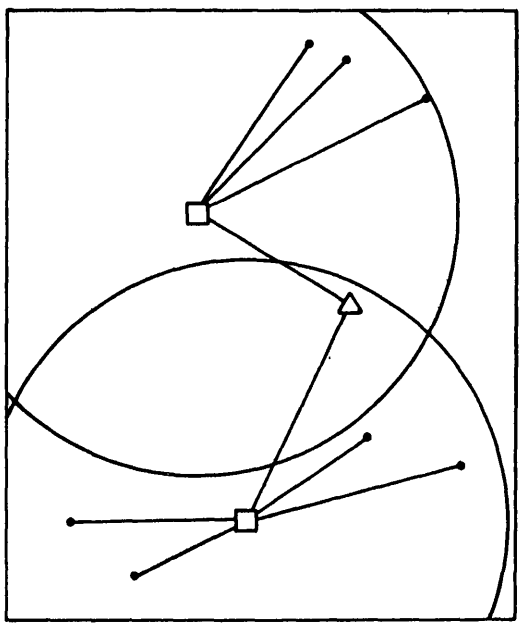

Fig. 4 — Example of a linked cluster organizational architecture

entire network and proceeds to form the clusters and to assign cluster heads. In fact, to facilitate the description, we shall further assume, for the moment, that the communication range is fixed and the same for all nodes. After describing the centralized versions of the cluster head selection rules, we shall explain their more interesting, distributed implementation, in which no node possesses any prior knowledge about the other nodes, no coordinating node is present, and the communication range is variable and unknown.

Before proceeding with the description of these algorithms, however, we introduce the following terminology for describing the linked cluster structure: (a) Two nodes are said to be *neighbors* if they can communicate with each other via a half-duplex HF channel. Thus,we do not consider one-way communication capability sufficient for connectivity. (b) Two clusters are said to be *directly linked* if their cluster heads are neighbors. (c) A node is a *member of a cluster* if the node is a neighbor of the cluster head or if the node is itself the cluster head. (d) A cluster *covers* another cluster if each member of the second cluster is also a member of the first cluster.

Cluster Head Selection Rule (LCA) — Centralized Version

This method produces the node clusters shown in Fig. 5a. The nodes are first numbered from 1 to N, where N equals 10 for the example shown in Fig. 5. The central controller starts with the highest numbered node, say node N, and declares it a cluster head. Then it draws a circle around that node N with radius equal to the range of communication. The nodes inside the circle form the first cluster. It then considers whether there are nodes outside this circle. If there are, it tentatively considers drawing a circle about node 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. Then consideration of tentative cluster head status for nodes N-2, N-3, etc. follows, until all nodes lie within at least one circle. The resultant arrangement provides every node with a cluster head. Any pair of clusters may be linked directly; they may even cover one another, overlap, or they may be disconnected. In the last two cases, selected nodes must serve as gateways for the interconnection of the cluster heads. This issue will be addressed in the discussion of the distributed version of the algorithm.

Cluster Head Selection Rule (ALCA) — Centralized Version

In the alternative method the procedure is a slight variation of the one just described. The central controller starts with the lowest numbered node, node 1, declares it a cluster head, and draws a circle

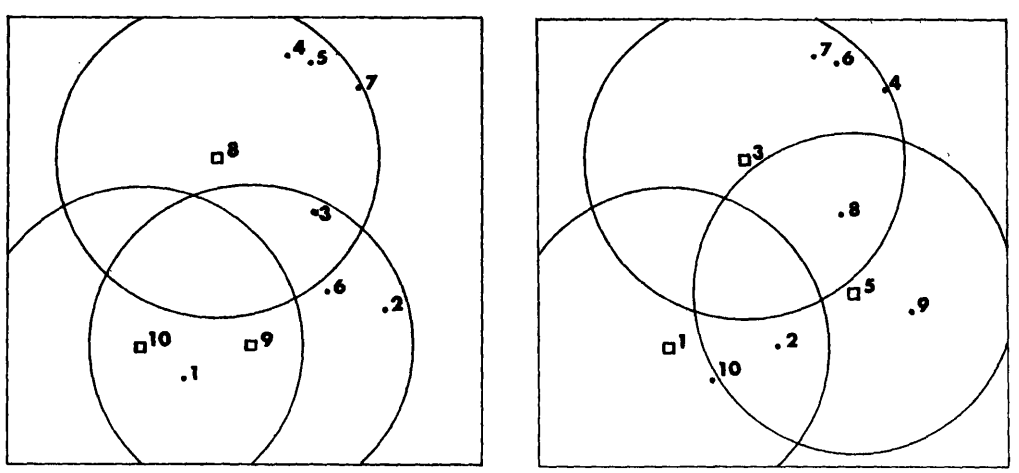

Fig 5 — Illustration of cluster formation using the LCA and ALCA cluster head selection rules

around it with radius equal to the fixed communication range, thus forming the first cluster. If node 2 lies in this circle it does not become a cluster head. If not, it does become a head and the controller draws a circle around it. Continuing in this manner; node i becomes a cluster head unless it lies in one of the circles drawn around earlier nodes. The resultant arrangement is shown in Fig. 5b. Unlike the previously described case for the LCA, with the latter method, no cluster can cover another nor can two clusters be directly linked. To facilitate comparisons with examples using the LCA, the nodes are numbered in reverse order from that shown for the LCA. Thus, nodes 1,2,3, etc. of the LCA examples are nodes N, N-1, N-2, etc. of the corresponding ALCA examples. This node numbering convention is used in the remainder of the present report.

We now describe in detail the distributed versions of these algorithms. The LCA is also described in Ref. 9.

The Linked Cluster Algorithm (LCA) — Distributed Version

The network architecture shown in Fig. 4 is not, by itself, adequate for the HF ITF network. The structure shown in Fig. 4 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. 4 and being based on a particular connectivity map. Moreover, these connectivity maps are continually being reformed in order to adapt to the time variation of the HF 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 to avoid the subband in which the network is being reorganized. Appropriate message framing provisions must be made in order to avoid interruption of message transmissions at the beginning of the corresponding reorganization epochs.

The algorithm 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 some simple message exchange is necessary. This exchange is assumed to take place over the control channel [3]. 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.

Cluster head and gateway nodes and the links that interconnect them form the backbone network. It is not clear whether all or only part of the intercluster traffic should be routed through this backbone network. This would depend on routing and delay considerations that have not yet been thoroughly examined. 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 to reduce the number of local controllers and relay steps in transmissions. On the other hand, superfluous links can provide path redundancy and also serve to deconcentrate intercluster traffic. We shall show how some path redundancies can be removed, if desired, by direct use of our algorithm. Further elimination of links can be achieved by separate use of other algorithms similar to the ones described by Dalal [10] and Gallager [11]. These issues, however, are not primary at this time and should not detract from the description of the algorithm. They are examined in greater detail later in this report.

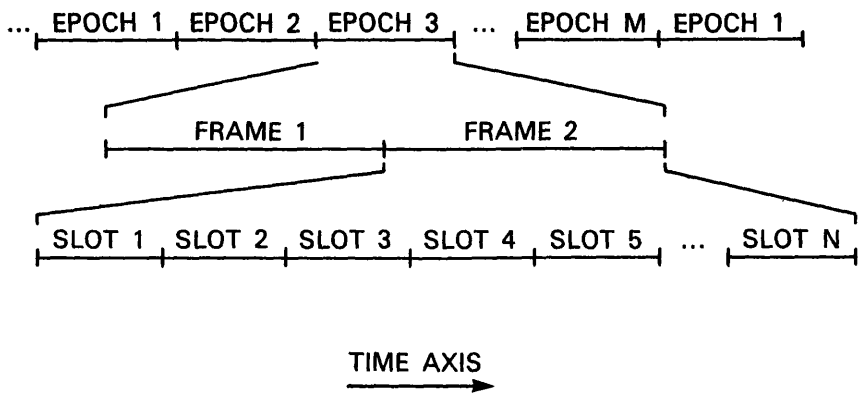

Fig 6 — Control channel schedule

The schedule of events in the algorithm is shown in Fig. 6. All nodes are assumed to keep accurate global time. The frequency band of the HF channel is partitioned into subbands. Within each subband the communication range of the ground wave is approximately constant. The width of such subbands is considered to be of the order of 1 MHz or larger. The precise way the HF band actually should be partitioned will depend on equipment constraints, frequency allocation regulations, and spread spectrum signaling limitations which will be discussed later in this report. It is not relevant to the algorithm as such. The algorithm runs separately for each frequency range since the connectivities depend critically on frequency for the HF band. Thus a different network organization is formed for each frequency range. However, the *networks* corresponding to the different frequency subbands are not actually autonomous, but rather they constitute a single HF ITF network.

The time required to complete the algorithm at a particular frequency range is called an *epoch.* The algorithm is repeated for each frequency subband and eventually returns to the original band and repeats the whole cycle. Each epoch is divided into two frames; each frame is subdivided into N time division multiple access (TDMA) slots, where N is equal to the number of nodes in the network. Each node is identified by an 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.

It is not essential in the LCA that the number of slots in each LCA frame equal the number of nodes in the network; the number of slots could be greater than or less than the number of nodes. For example, LCA transmission slots are not needed for submarines that maintain radio silence As another example, there may be so many nodes in the network that the LCA frame length would be too long if all nodes were to be included in the LCA transmission schedule. In the latter case, only a subset of all the nodes would participate in setting up the backbone network. Of course, all nodes must be provided access to the backbone network for the purpose of intercluster communication.

An alternative method for implementing self-organizing mobile networks, which does not use preassigned slots, is described in Ref. 12. A disadvantage of the latter approach is that, as the contention for unused slots increases, it takes increasingly longer for the network to organize.

The staggering in the time domain of the epochs for the different subbands provides the needed robustness against topological changes as it allows the ITF network to adapt to connectivity changes in one subband while maintaining regular communication in all other subbands Through continual monitoring of possible changes, the network has the capability to maintain connectivity. During each epoch, regular communication is allowed to take place at all other frequencies except the one corresponding to that epoch. Instead, at that frequency, only control messages are exchanged that permit the organization of the network. Thus the overhead encountered due to the proposed algorithm is equal to $1/M$

where M is the number of frequency ranges into which the HF band is divided (for example, 1/28 if we consider 1 MHz subbands of the 2-30 MHz HF band).

During execution of the LCA each node maintains the following data structures for each subband: 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 bidirectional 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 intent of the algorithm is to enable each node to acquire such a partial version of the global connectivity matrix, which is of course unknown.

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; for example, reception at a node may be poor due to jammer proximity. A built-in feature of our algorithm is the capability to distinguish between unidirectional and bidirectional connectivities.

The HEADSONEHOPAWAY list is a list of all the cluster heads that are connected to a node and, finally, the HEADSTWOHOPSAWAY list identifies 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.) By the end of this frame, node i has filled in some of the entries of its connectivity matrix. In particular, it can fill in the elements above the main diagonal, i.e., the elements (i, j) of the ith row that satisfy $j > i$. (At the start of each epoch, the connectivity matrix is initialized to the identity matrix.) The element (i, j) is set equal to 1 if:

(a) node i heard from node j and
(b) 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. By the end of the frame each node knows the two-way connectivities for itself and for its neighbors. The global connectivity matrix is not available to every

individual node — only a partial version of it is formed by each node. However, for the case of error-free transmissions, all versions are consistent with the global true matrix. Later on we consider the effects of transmission errors.

Formation of Clusters — At the ith slot of the second frame, node i can perform a logical function that permits it to evaluate its current NODESTATUS (i.e., whether it is a HEAD or an ORDINARY node), which it can then transmit along with its row connectivity. 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 own connectivity row. If there is no neighbor with higher identity number, node i becomes a cluster head. If another neighbor exists with higher identity number, that neighbor will become a cluster head, so i doesn't 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 is able to broadcast in the ith slot of the second frame his NODESTATUS. This information is needed for the linking of the clusters as will be seen later in this report. 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 its NODESTATUS prior to its Frame 2 transmission. Consequently, an additional frame would be needed for the exchange of NODESTATUS information. (See for example, an earlier version of the Linked Cluster Algorithm [13].)

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. This list is complete since each node that becomes a cluster head broadcasts its NODESTATUS during its Frame 2 transmission. For the second list, the node examines each of its neighbors. By looking at a neighbor's connectivity row it can identify that neighbor's cluster head. If that cluster head is not within one hop of this examining node, the cluster head is put into the HEADSTWOHOPSAWAY list. It is possible that a two-hops-away neighbor of a node has become a cluster head unbeknown to that node; an example of an undetected two-hops-away head is given in the third section of this report. The HEADSTWOHOPSAWAY list is therefore not necessarily complete, unlike the HEADSONEHOPAWAY list. Thus, by the end of the second frame, clusters are formed and the data base necessary for the second logical function of the algorithm (the linking of the clusters) is available.

The previously described procedure may produce a few unnecessary cluster heads under some circumstances. An intermediate, auxiliary procedure, called DELETEHEADS, can be used to eliminate the truly redundant clusters that may have been formed. Let us recall that one cluster covers another if all the nodes of the second are also members of the first. It is straightforward to verify that the cluster forming procedure will never produce clusters mutually covering each other. However, it is possible for one cluster to cover another. In that case, DELETEHEADS can take over and eliminate the covered, and thus unnecessary, cluster head. Every node in the clusters involved can detect this coverage since its database includes the connectivity rows of all of its neighbors and may proceed to adjust its data base by returning the head of the redundant cluster to ordinary NODESTATUS and by deleting the covered head from the HEADSONEHOPAWAY list if it is a neighboring node.

Linking of the Clusters — The next logical stage is the linking up of the clusters. This is accomplished by the introduction of GATEWAY nodes. Every nonhead node is a candidate to become a GATEWAY. There are three cases to be considered; these are illustrated using a simple, fixed range communication model. The first case is shown in Fig. 7a. Here there is no need for gateways since the heads of the clusters are directly linked. (This case does not occur in the ALCA.) The second case is depicted in Fig. 7b. Here exactly one 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. 7c. Here

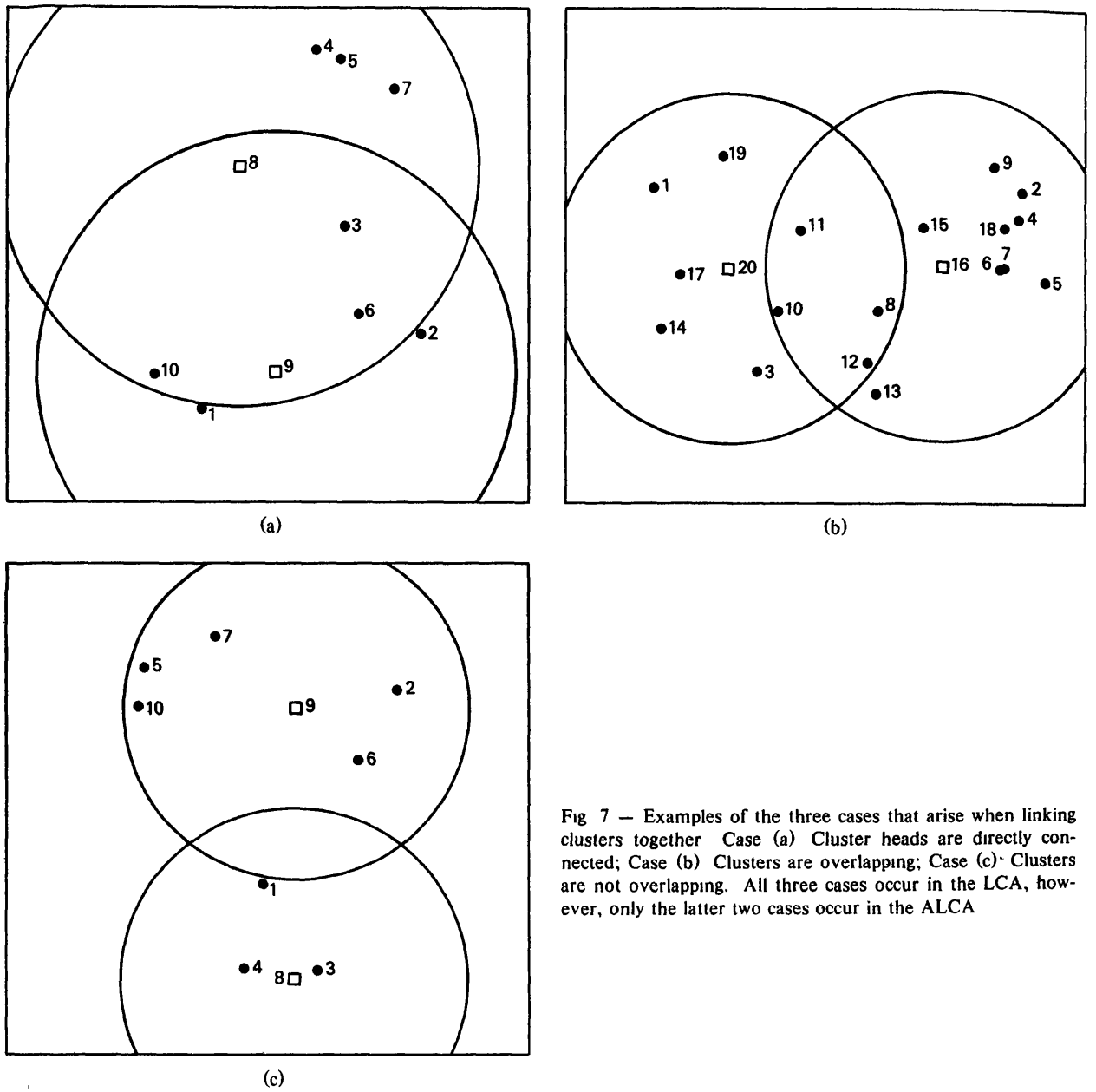

Fig 7 — Examples of the three cases that arise when linking clusters together Case (a) Cluster heads are directly connected; Case (b) Clusters are overlapping; Case (c) Clusters are not overlapping. All three cases occur in the LCA, however, only the latter two cases occur in the ALCA

at least two nodes (one from each cluster) are needed. It is, of course, assumed that suitable such nodes do exist; otherwise, the network cannot be connected. We now describe the procedures used to achieve the linkup in the last two cases.

Case of overlapping clusters — This case is handled by procedure LINKUP1. Pseudocode for LINKUP1 is given in the Appendix of this report. Every node that is not a cluster head is a candidate gateway node. Each pair of heads in a node's HEADSONEHOPAWAY lists corresponds to a pair of overlapping clusters. To avoid the formation of unneeded gateways, candidate gateway nodes first test to see if the heads of overlapping clusters are already linked through another cluster head. An example of such a linkage is examined in detail in the third section of this report. (The test for linkage through another head need not be performed when using the ALCA since cluster heads are never directly linked in that case.) 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 and every node possesses the connectivity row information for every one of its neighbors. Thus there is no ambiguity in the selection of the gateway node.

Case of nonoverlapping clusters — This case is handled by procedure LINKUP2. Pseudocode for LINKUP2 is given in the appendix of this report. Cluster head pairs consisting of a node's own head plus a head from the node's HEADSTWOHOPSAWAY list identify nonoverlapping clusters. (The clusters may in fact be overlapping, unbeknown to the examining node). 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 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 created via LINKUP1 through another cluster head. Thus it seeks nodes among its neighbors that may include in their connectivity rows the second member of the head-pair it is examining, together with a cluster head from its HEADSONEHOPAWAY list. In other words, if a head in its HEADSTWOHOPSAWAY list is connected via one of its neighbors to a cluster head that was earlier connected to the first member of the pair via LINKUP1, there is no need for additional linkage. If this circumstance is not established, the node proceeds further with the LINKUP2 procedure, assuming that its services are needed for this linkage. 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 extra gateways with two or possibly more pairs becoming gateways. It is worth noting, however, that such multiple linkage outcomes are not very likely for most topological configurations. In some cases 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 and known to it. 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, however, are rare and only a harmless nuisance. They need not affect the network's operation and cannot be avoided without substantially increasing the data bases available at each node by additional message exchanges.

The Alternative Linked Cluster Algorithm (ALCA)-Distributed Version

The two algorithms have nearly identical implementations. Both use the same data structures, and both follow the same transmission schedule shown in Fig. 6. Also, the formats for the control messages are nearly the same for the two algorithms; the only difference is that the rules for forming the clusters and for assigning cluster heads are different. The difference has already been described in the centralized version. In the distributed implementation, the rule is simply that a node becomes a cluster head unless it has a lower numbered cluster head for a neighbor. Thus instead of announcing its node status, each node broadcasts the identity of its own cluster head during ALCA Frame 2 transmissions. This enables nodes to fill in both their HEADSONEHOPAWAY and HEADSTWOHOPSAWAY lists. One additional, difference exists between the two algorithms; namely, in the ALCA, gateway node pairs are selected from the lowest value links instead of the highest. Thus in the ALCA, the lower numbered nodes are more likely to become cluster heads and gateways whereas the LCA favors the higher numbered nodes for these roles.

The advantage of the ALCA is that it is simpler to implement than the LCA. Because of the particular cluster head selection rule used in the ALCA, no superfluous (i.e., covered) clusters are formed. Thus procedure DELETEHEADS, which is a rather involved procedure used in the LCA, is not required by the ALCA. The ALCA also gives a more uniform distribution of cluster heads. On the

other hand, the LCA generally yields network structures that contain fewer cluster heads, gateways, and backbone network links.

THE SIMULATION MODEL AND SIMULATION RESULTS

A simulation model was constructed to provide examples of network structures obtained with both algorithms. This model represents the first phase in the development of a comprehensive HF ITF network simulation facility. The development of the HF ITF network simulator is proceeding alongside the design of the network itself. The simulation model is coded in the general purpose, high level language SIMULA [14] and runs on the Laboratory's DEC-10 computer. The ITF network simulator uses some of the SIMULA software tools developed for the simulation of satellite networks [15-17].

The user can either enter node position data directly into the simulator or let the simulator itself randomly position the nodes. Random positioning of nodes was used for nearly all of the examples shown in this report. In the latter case, the user must specify the number of nodes and the radius R of the circular area within which the nodes are to be located. For the examples shown in this report, R equals 300 km.

The placement of nodes is based on a uniform distribution of nodes over the task force area. Thus a node is equally likely to assume any azimuthal position. On the other hand, the radial distribution of nodes is nonuniform. The cumulative distribution function $F(r)$, which gives the probability that a node is positioned within a radius r from the "center" of the task force, is given by the ratio of the circular areas of radii r and R; thus

$$F(r) = r^2/R^2$$

Based on the preceding considerations, the simulator uses the following procedure to position the nodes. Beginning with node 1, the radial distance $r(1)$ of this node from the center of the task force is obtained from

$$r(1) = \text{R*SQRT(UNIFORM(0,1,SEED))}$$

where UNIFORM is a SIMULA-provided procedure that returns a random number in some specified range (0 to 1 in this case). The random number is chosen from a uniform distribution. The SEED for generating this random number is changed as a result of each call to UNIFORM. Next, the relative azimuthal position of node 1 is determined by the formula

$$\text{THETA(1) = 2*PI*UNIFORM(0,1,SEED).}$$

Similarly, the positions of nodes 2, 3, ..., N are determined, in sequence, according to the procedures just described. For a very large number of nodes, these procedures tend to provide a near-uniform distribution of nodes over the area of the task force.

In our simulation model, the determination of whether two nodes are within communication range is based on the high-frequency groundwave range curve shown in Fig. 8. This curve can be expressed as a combination of polynomials of the form

$$a0 + a1*f + a2*f^2 + a3+f^3 + a4*f^4$$

where f is the frequency in MHz and the values of the coefficients are given in Table 1. The curve shown in Fig 8 is an approximation to the global worst-case noise limited nighttime groundwave range performance curve given in [18]. The curves assume 1 kW transmitters. Reference 18 explains further details regarding the assumptions used in deriving this curve. The groundwave range curve of Fig. 8 is only intended as a model. Actual ranges will vary with geographic location, sea state, time of day, season, and threat levels (i.e. jamming).

Table 1 — Polynomial Coefficients for Groundwave Propagation Model

Frequency Range [MHz]	a0	a1	a2	a3	a4
2 to 10	75.83	219.512	−55.4377	5.449301	−.192016
10 to 20	216.27	33.623	−4.8438	0.224165	−.003497
20 to 30	33.60	34.198	−2.3805	0.064103	−.000583

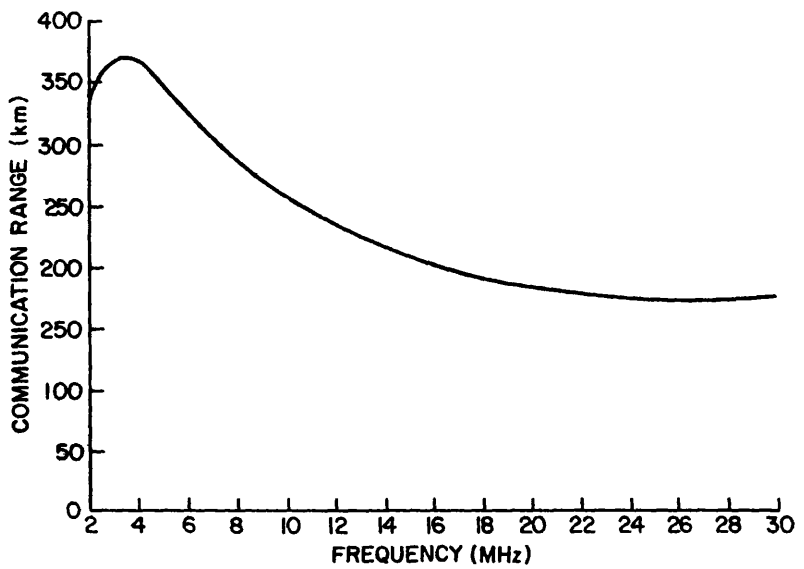

Fig 8 — Groundwave communication range model used in simulator

The global connectivity matrix for frequency f consists of entries 0 and 1 where entry $[i, j] = 1$ if node i is in range of node j, as determined by Fig. 8, and entry $[i, j] = 0$ otherwise. Note that in this simple model all connectivities are bidirectional; thus, the global connectivity matrix is symmetrical.

We now turn from the description of the simulation model to a discussion of results obtained with it. We specifically consider the following features: backbone network complexity, undetected cluster heads, frequency dependence, node numbering methods, deadend loss of links, cyclic paths, loss of nodes, and loss of links.

Backbone Network Complexity

An interesting characteristic of our linked cluster structuring algorithms is the complexity of the resultant backbone network. Two quantities that can be used to quantify this complexity are, for a given total number of nodes, the number of clusters formed and the number of links in the backbone network. Let us assume that two backbone networks A and B are of equal complexity if one of the following is true:

- A and B have the same number of cluster heads and the same number of links,
- A has more heads but fewer links than B,
- B has more heads but fewer links than A.

If two backbone networks are not of equal complexity, then the one with the least number of clusters or links is said to be the simpler network. There are cases when the ALCA gives backbone networks that are either more complex, less complex, or of equal complexity to those obtained with the LCA Examples of the occurrence of these three cases, for the same distribution of nodes but for different frequencies, are shown in Fig. 9. A comparison of examples in Fig. 9b shows that, in some cases, the two algorithms produce the same backbone network. The histogram of Fig. 10 compares the complexities of the backbone networks obtained with the LCA and ALCA for 72 simulation runs. These data indicate that there is a tendency for the LCA to produce slightly less complex backbone networks. It is not clear, however, whether this is an advantage or disadvantage for the overall performance of the network.

Undetected Two-Hops-Away Heads

Each node keeps a list of those heads that are one hop from it and those that are two hops away. The former list is always complete, but the latter may be incomplete. An example of an undetected two-hops-away head occurs in Fig. 9b. In this figure (see ALCA case), node 3 is unaware that node 7 is a head. For node 3 to detect that 7 was a head, either node 16 or 20 would need to have 7 as its cluster head; however, node 3 is the cluster head for both 16 and 20.

Frequency Dependence

Since the communication range varies significantly across the high-frequency band, we envision the network as consisting of the overlay of several sets of linked clusters, each set derived from a connectivity map formed using a different frequency. The frames of Figs. 11 and 12 show the resultant network structures for six epoch frequencies and for the LCA and ALCA. The connectivity maps for these frequencies are shown in the corresponding frames of Fig. 3. The particular frequencies used were chosen because they provide examples corresponding to a wide variation in the communication range. In these examples, connected networks are formed at all but the highest frequency.

The technique of overlaying several sets of linked clusters provides alternative communication paths. If a backbone network link is lost at one frequency due to jamming, other backbone networks at other frequencies can be used. When the network is reorganized in the subband in which the jamming occurs, a new backbone network will be set up that will not contain the jammed link. The combination of path redundancy and the ability of the network to adapt to node and link losses results in a robust ITF network.

Unfortunately, because the same nodes tend to become heads and gateways for each epoch in the examples of Figs. 11 and 12, these nodes will be overburdened with network management and traffic direction responsibilities. Moreover, the appearance of the same nodes in several different backbone networks makes the network too dependent on these nodes. For example, in Fig. 12, the loss of node 2 would sever the backbone network for all but the lowest frequency. Although the network would begin to compensate for the loss of this node by restructuring the backbone network epoch by epoch, parts of the network might remain disconnected until a full cycle of epochs occurred. This problem can be avoided by introducing a dynamic node numbering strategy.

Node Numbering

Given the simple strategy used in deciding the identity of a cluster head or a gateway node among a group of candidates, it is clear that number assignment to the nodes is a very important part of the proposed organization. For example, in the ALCA the lower numbered nodes simply have a greater tendency to become heads or acquire gateway status than higher numbered nodes while in the LCA the opposite is true. If the numbering system is the same for all epochs, these nodes will be overburdened

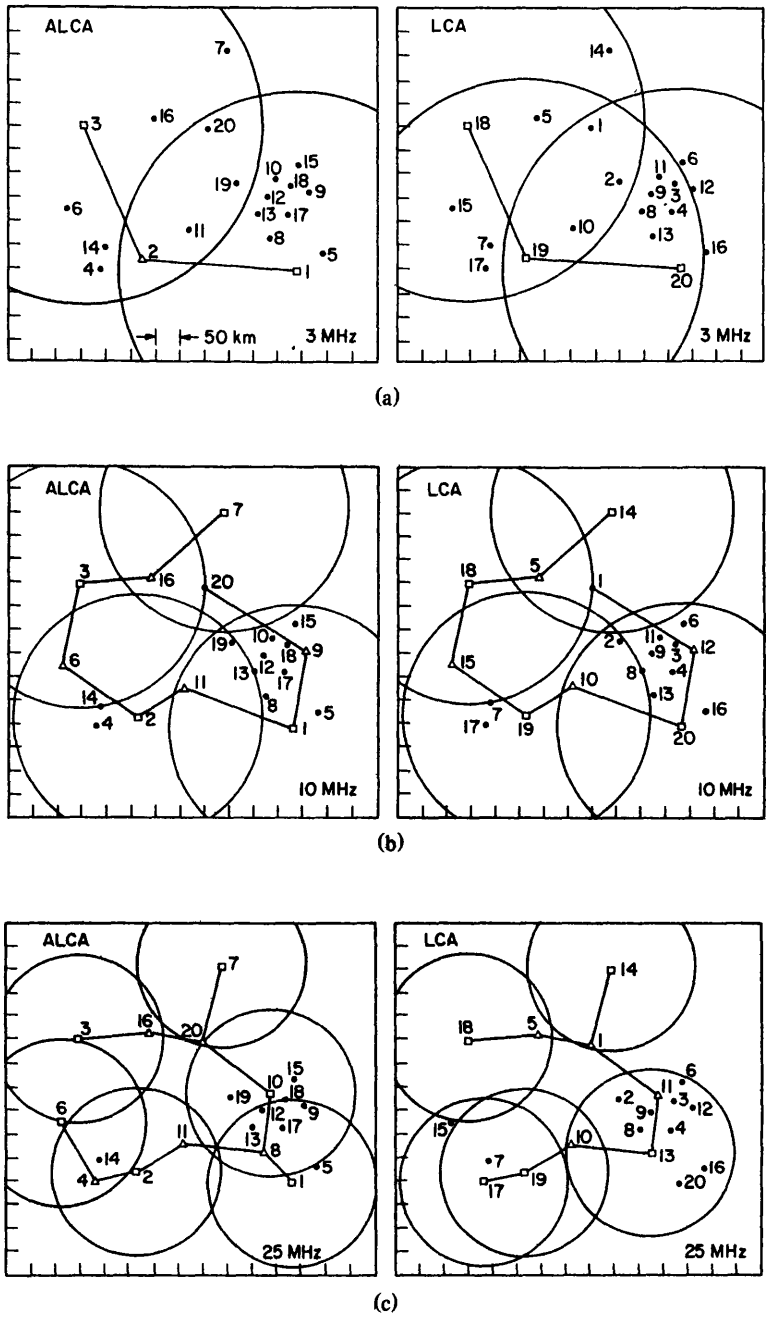

Fig 9 — Three cases that arise when comparing the relative complexities of the backbone networks obtained with the LCA and ALCA Case (a) The structure obtained with the ALCA is the simpler, Case (b) The backbone networks are of equal complexity, Case (c) The structure obtained with the LCA is the simpler

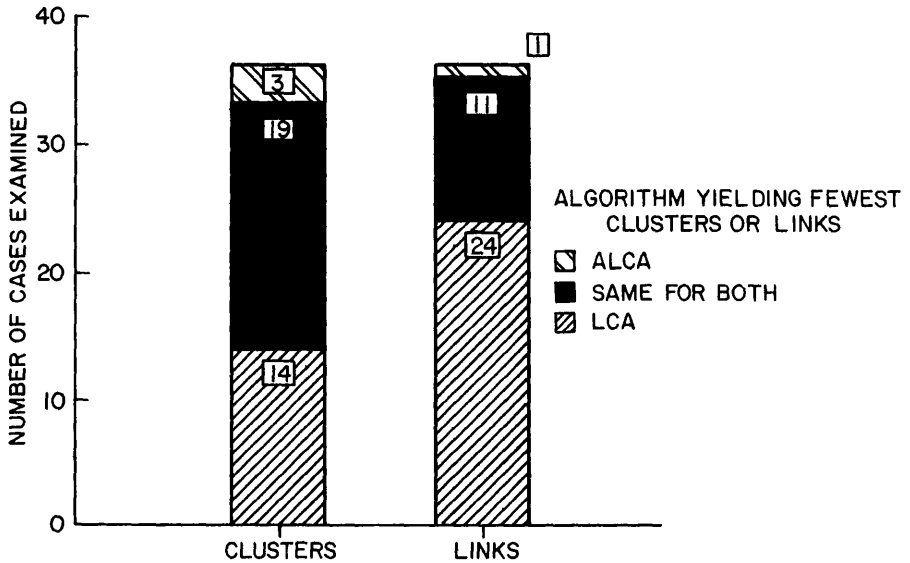

Fig. 10 — Histogram comparison of which algorithm (LCA or ALCA) yields backbone networks with the fewer number of clusters and the fewer number of links

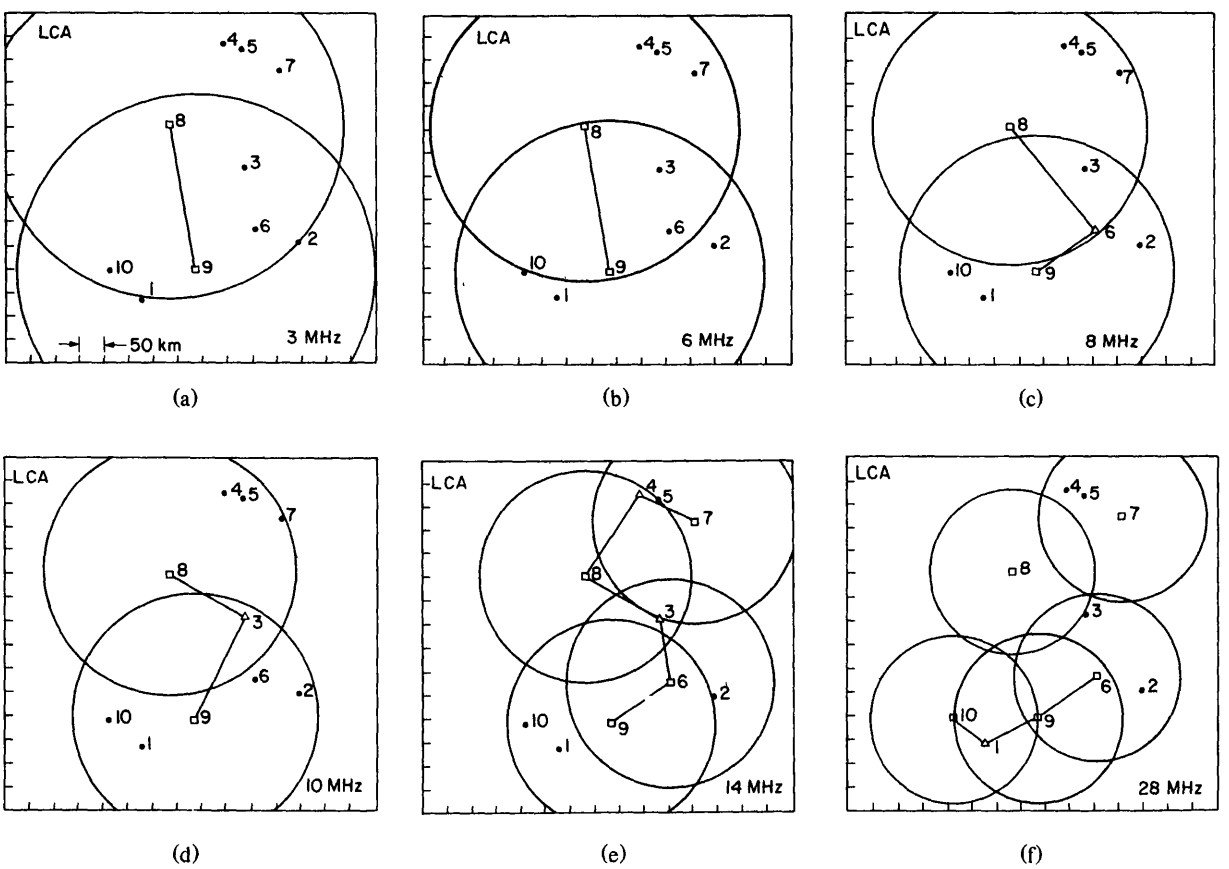

Fig 11 — Network structures obtained with the LCA for a case with six different epoch frequencies

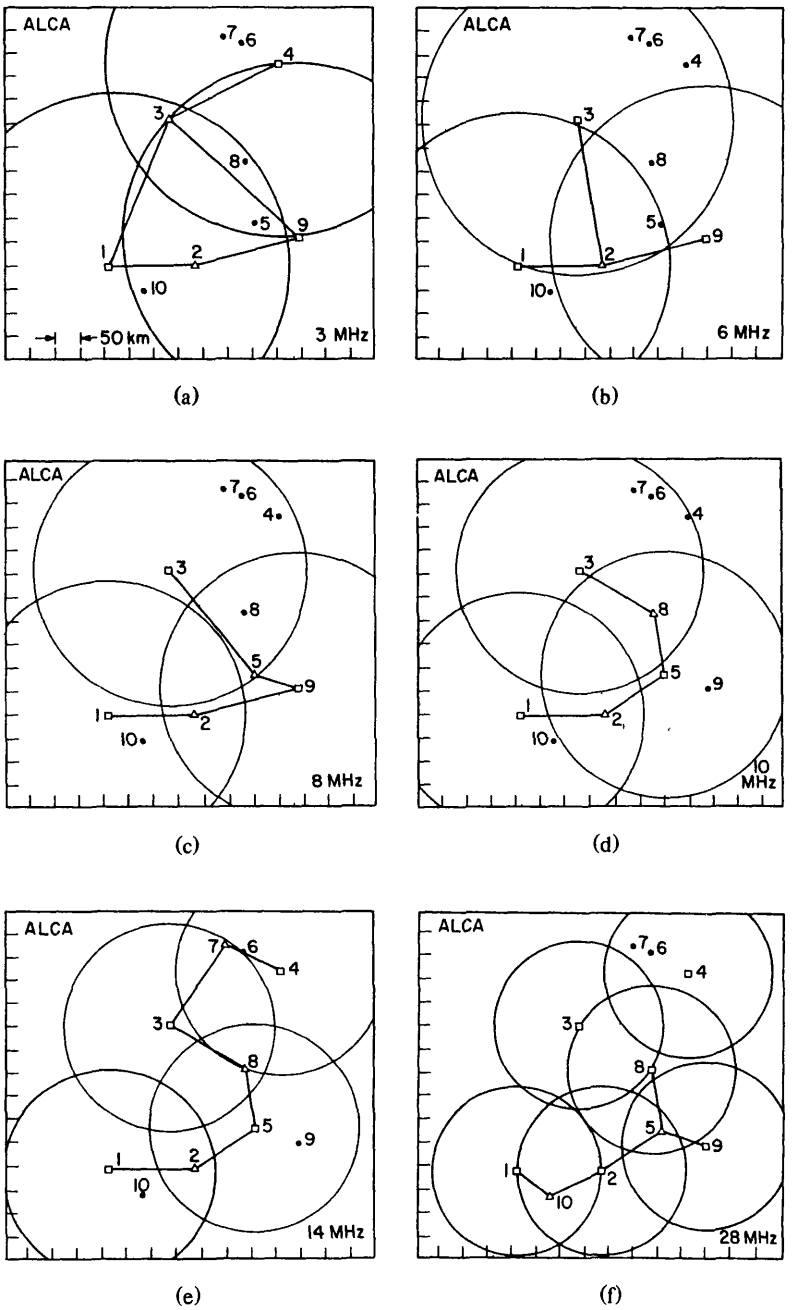

Fig 12 — Network structures obtained with the ALCA for a case with six different epoch frequencies

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 epoch. Also, if all nodes do not have comparable equipment, favored numbers should be assigned to nodes with greater capability. A simple strategy that tends to produce disjoint backbone networks, i.e. having no nodes or links in common, is to invert the numbering on successive epochs. That is, nodes 1, 2, 3, etc. become nodes N, N-1, N-2, etc. An example of such a strategy is shown in Fig. 13 for both the LCA and ALCA. The results show some separation of the backbone networks, however, the nodes numbered 3 and 9 in Frames (a) and (d) and 8 and 2 in Frames (b) and (c) still appear in each of the backbone networks. That this is unavoidable can be seen by considering the complete connectivity map for this particular example which appears in Fig. 3d. Since nodes 2(9) and 8(3) are cut set nodes, they must necessarily be part of the backbone network. In general, a strategy of node number inversion followed by number randomizing on alternate epochs should produce well separated backbone networks if there are no cut set nodes. Such a strategy could, for example, give fixed but different node numbers for each subband. Numerous other renumbering strategies are of course possible. If, despite frequent node renumbering, a node becomes a cluster head or gateway node in several subbands, then this node is likely to be a critical node in the sense that its loss may disconnect the network.

Dead-End Links

Because each node has knowledge of the network out to only two hops from itself, certain inefficiencies can arise in the selection of gateways and links for the backbone network. One manifestation of this inefficiency is the creation of dead-end links in the backbone network. By a dead-end link we mean a link of the backbone network that connects a gateway node with an ordinary node. These links form as a result of attempts to link some nonoverlapping clusters. Three examples of networks with dead-end links are shown in Fig. 14; each of these examples illustrates a different underlying cause for the creation of the dead-end link.

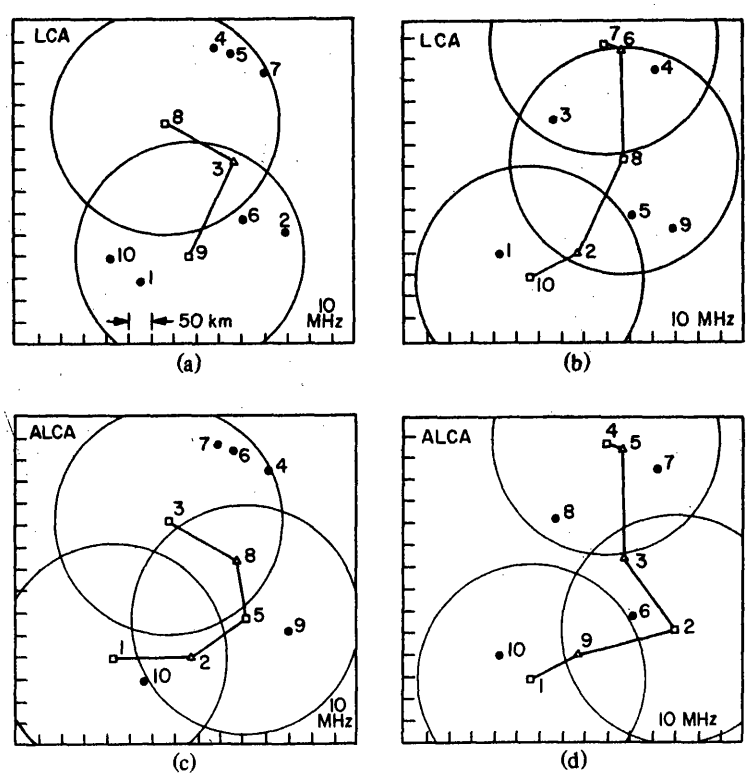

Fig. 13 — Examples, using both the LCA and ALCA, showing the results of node renumbering

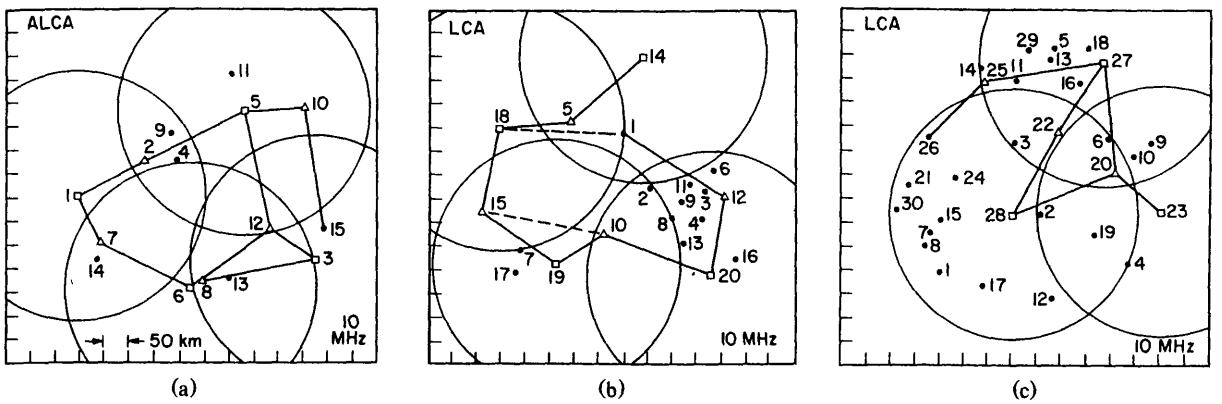

Fig 14 — Examples of dead-end links The cause of the dead-end link is different for each of the examples shown

In Fig. 14a, node 10 believes that it must become a gateway, along with node 15, to link nonoverlapping clusters 3 and 5. Because node 10 is out of range of node 12, node 10 is unaware that (a) 12 is connected to 3 and 5, (b) clusters 3 and 5 are in fact overlapping, and (c) that cluster heads 3 and 5 are linked via node 12. On the other hand, 15 does know that clusters 3 and 5 overlap, and hence, 15 does not form part of the gateway pair (10,15).

A different scenario developed in the case of Fig. 14b. In this case there are four cluster heads, 20, 19, 18, and 14. Node 15 lies in the intersection of clusters 18 and 19 and is a gateway linking them, as is node 10 for clusters 19, 20 and node 5 for clusters 18 and 14. Node 10, in its search to link nonoverlapping clusters, formed a potential pair of gateways with node 15 for link clusters 18 and 20. Now consider what happens in the case of nodes 1 and 12 in their attempt to link 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 12 is not aware of this. Thus, node 1 computes both sums (1 + 12 = 13 and 15 + 10 = 25) and determines that its services as a gateway are unneccessary, 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 a dotted line). Hence, the creation of the dead-end link (1,12). Actually, although link (10,15) is a prime candidate for joining together nonoverlapping clusters 18 and 20, this link does not become part of the backbone network since cluster heads 18 and 20 are already linked via nodes 10 and 15 and cluster head 19. We shall discuss this point again, in greater detail, later in this section when we consider methods for avoiding the formation of some cyclic paths in the backbone network.

The example shown in Fig. 14c is a consequence of the use of procedure DELETEHEADS in the LCA and does not occur when the ALCA is used. In this example, five cluster heads are initially created before the call to procedure DELETHEADS, namely, nodes 30, 29, 28, 27, and 23. However, cluster 30 is covered by 28 and 29 is covered by 27; thus, only three clusters appear in Fig. 14c. Now consider what happens at node 25. Node 25, which originally had 29 as its cluster head, is reassigned to cluster 27 which covers 29. Because node 25 is outside of the original cluster 30, it is unaware that 30 is covered by 28. Hence, node 25 will consider whether it should become part of a gateway pair linking non-overlapping clusters 27 and 30. Node 25 will compute that the link (25, 26) (value: 25 + 26 = 51) is the preferred link connecting these two clusters. Thus 25 becomes a gateway node. On the other hand, node 26 does know that node 30 is covered by 28 and also that 27 and 28 are already linked Thus, 26 does not become part of the (25,26) gateway pair and there results a dead-end link.

Since gateways have only a passive relaying function, in contrast to the cluster heads, which take an active controller's role, no harm is done by the occurrence of these extraneous gateways and links.

However, it does represent a weakness of the algorithms, a typical one for the case of distributed data bases. The elimination of such superfluous, half-way links is desirable, but the matter is not pursued further in this report. It will be reexamined when the issue of routing is addressed in detail.

Cyclic Paths in Backbone Network

Our network structuring algorithms allow each node to determine for itself whether it should become a cluster head or gateway, or remain an ordinary node. Heads and gateways compose the nodes of the backbone network. The backbone network links are those links that connect neighboring cluster heads plus those links connecting gateways and the nodes they seek to join together. We have given numerous examples of backbone networks. In some of these examples, the backbone network contains cyclic paths. Here, we consider how and why these loops form and show how our network structuring algorithms prevent the formation of certain types of loops.

Algorithms exist for obtaining loop free networks [10, 11], but the use of these algorithms in the ITF network would result in significantly increased communication overhead. On the other hand, loops can provide alternative paths between some nodes. Our approach in designing the network structuring algorithms has been to simplify the structure of the backbone network by preventing the formation of some types of loops without increasing the communication overhead. Path redundancy can be obtained by forming several different backbone networks. Current versions of the LCA and ALCA check for, and prevent, the formation of loops in two cases. The first case occurs during the linking of overlapping, but not directly connected, clusters and is illustrated by the (contrived) LCA example shown in Fig. 15. The second case can occur when non-overlapping clusters are linked and is illustrated by the example of Fig. 14b.

The structure shown in Fig. 15 evolves as follows: node 5 becomes a head because it is the highest numbered node; node 4 becomes a head because it is the highest numbered node within the circle centered about itself; node 2 becomes a head because it is the highest numbered node connected to node 1. Links (2, 5) and (2, 4) become backbone network links since they link neighboring heads. Consider what happens at node 3. This node is not a head but it is a prime candidate for linking heads 4 and 5 because it is the highest numbered neighbor of these two heads. If 3 does become a gateway however, a loop will have been formed in the backbone network. We have included in the LCA a check for such loops; thus node 3 does not become a gateway. The check works as follows. A node that is a candidate for becoming a gateway linking overlapping, but not directly linked, clusters first checks to see if the cluster head pair is already linked by a third head. Note that in our example, node 3 can determine that heads 4 and 5 are linked by head 2 since node 3 is connected to each of these heads and nodes announce, during Frame 2, whether they are heads. But if node 3 were located in the shaded region of the intersection of circles 4 and 5, then 3 would not know that 2 was a cluster head and 3 would thus become a gateway. Note that situations such as shown in Fig. 15 cannot occur in the ALCA version because in that version cluster heads are never directly linked.

The utility of the checks performed by procedure LINKUP2, which prevent the formation of certain types of loop paths in the backbone network and which are described in the second section of this report are illustrated by the example of Fig. 14b. In this example, four clusters formed, having nodes 14, 18, 19, and 20 as their heads. Procedure LINKUP1 provides the following linkages: heads 14 and 18 are linked via gateway node 5, heads 18 and 19 are linked via 15, and heads 19 and 20 are linked via 10. If procedure LINKUP2 contained no checks for the creation of loop paths in the backbone network, then an additional link (10, 15) would become part of the backbone network. This would serve to join together nonoverlapping clusters 18 and 20.

To see how the checks built into LINKUP2 function, we examine the workings of the LCA at nodes 10 and 15. Both of these nodes know that link (10, 15) is a prime candidate for linking clusters 18 and 20. Node 10 determines that link (10, 15) is superfluous as follows. Node 10 finds that 19,

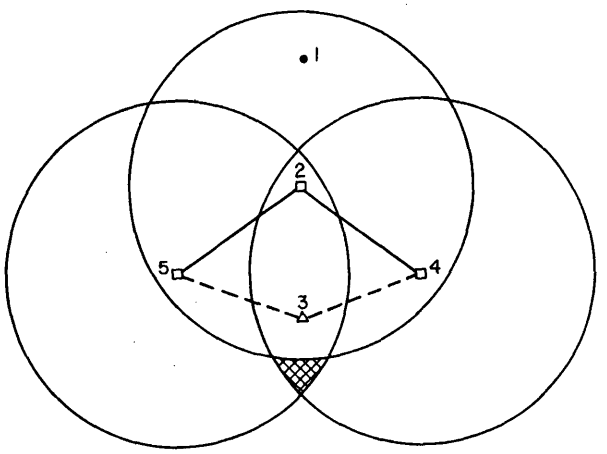

Fig. 15 — Example of the potential for forming a loop path when linking together overlapping clusters. Here, node 3 should not become a gateway. If node 3 is in the shaded region, however, it will become a gateway.

which is a head, is already connected to 20 because they are overlapping clusters; hence, they have already been linked by procedure LINKUP1. Furthermore, clusters 18 and 19 are overlapping and thus also linked. Hence, 10 concludes that a path already exists between heads 18 and 20 via 19, and therefore link (10, 15) is unnecessary. Likewise, node 15 will perform a similar analysis and also reach the conclusion that link (10,15) need not be part of the backbone network.

Several other types of loops can form in the backbone network. The formation of some of these loops could be prevented by minor additions to the LINKUP procedures. An example of a loop path that could easily be prevented from forming is shown in Fig. 14c. In this example, gateway node 20 could prevent the formation of loop path (20, 28, 22, 27, 20) by deleting link (20, 27) or (20, 28). Other loop paths may be so large that they go undetected during the exchanges of information in the LCA and ALCA. An example of such a loop is shown in Fig. 14a.

Loss of Nodes

Since the ITF network is a military network, its nodes may be disabled or destroyed by physical attack. Consequently, both of our network structuring algorithms provide for sensing node losses and for reconfiguring the network, if necessary. Examples of this are seen in Figs. 16 and 17 using the LCA and ALCA. Frame (a) of each figure shows the resultant backbone network for an initial network of 20 nodes. Each subsequent frame corresponds to a network obtained by deleting five nodes from the network shown in the preceding frame. The nodes that are *lost* are chosen from among the nodes that are most likely to become cluster heads or gateways. Since the LCA favors the selection of the higher numbered nodes as heads and gateways while the ALCA favors the lower numbered nodes for these roles, the five highest numbered nodes were deleted in successive frames for the LCA while the five lowest numbered nodes were deleted in successive frames for the ALCA. Because of the sequence in which our simulator model assigns node positions, corresponding frames of Figs. 16 and 17 contain the same set of nodes but with inverted numbering. That is, nodes 1, 2, 3, etc. of Fig. 16 correspond, respectively, to nodes 20, 19, 18, etc. of Fig. 17. This numbering scheme makes it easier to compare the results of the two algorithms.

Surprisingly, the LCA and ALCA give the same backbone network for each of the corresponding frames shown in Figs. 16 and 17, with the exception of the first frame. Using either algorithm the original network of 20 nodes, shown in Frame (a), is provided with a single, connected backbone network. Even with the loss of five key nodes, a single, connected backbone network is formed; see Frame (b) of either figure.

For the example shown in Fig. 16, the network adapts to the loss of nodes 16 through 20 as follows. The cluster head roles of nodes 17 and 20 (Frame (a)) are taken over by new heads 14 and 11

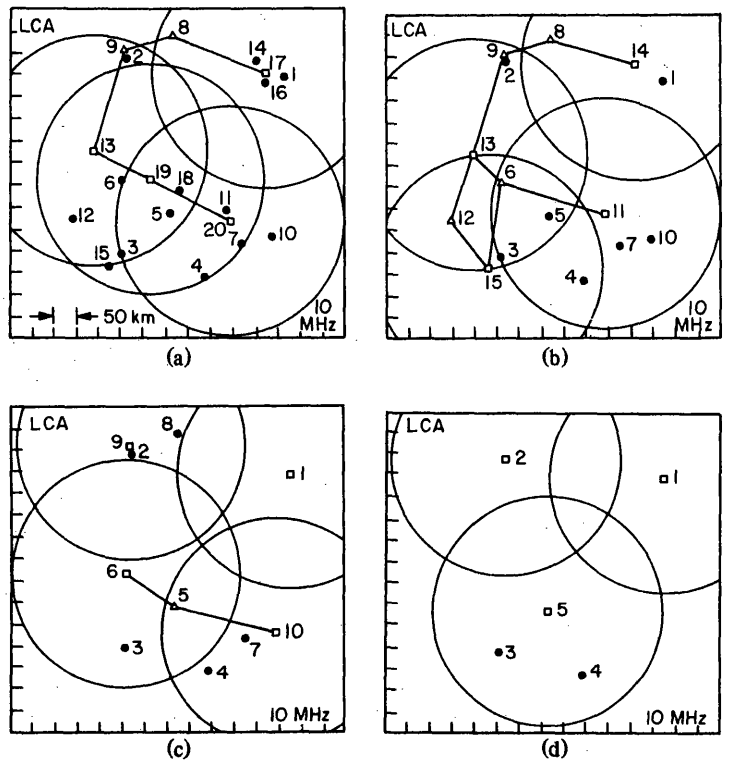

Fig. 16 — The effects of a loss of nodes using the LCA

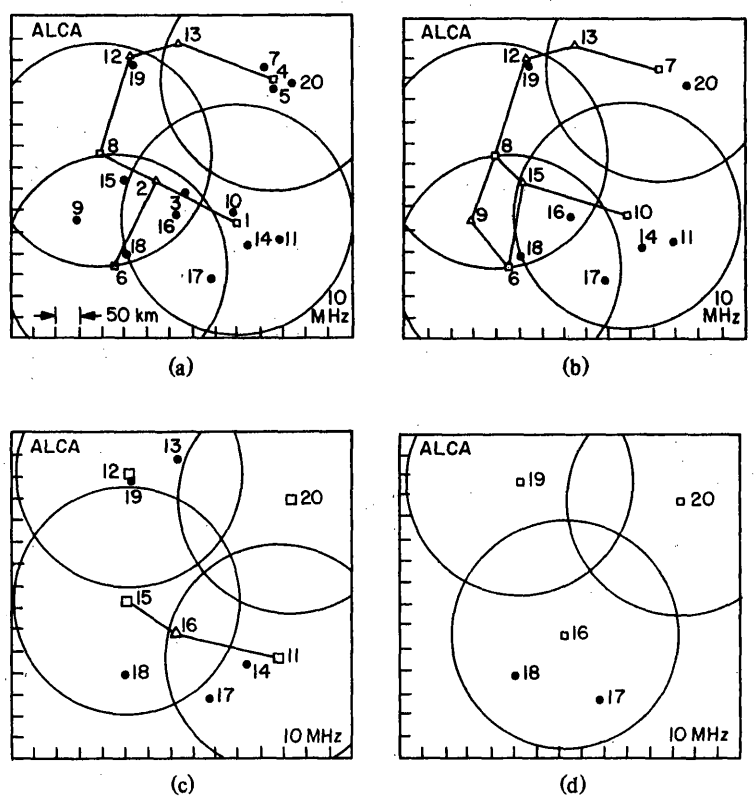

Fig. 17 — The effects of a loss of nodes using the ALCA

(Frame (b)). The loss of node 19 results in the creation of a new head at node 15, which is connected to cluster 13 via new gateway node 12 and to cluster 11 via new gateway 6. The role played by 19 in linking clusters 13 and 20 (Frame (a)) is now performed by gateway node 6, which links cluster 13 and new cluster 11. Nodes 16 and 18 are not part of the original backbone network so their loss is not of immediate concern.

The effects of the loss of nodes 1 through 5 in the case of the ALCA (Fig. 17) is quite similar to that just described for the LCA. In the case of the ALCA, the roles of heads 1 and 4 are taken over by nodes 10 and 7. The role of gateway node 2, which links clusters 1 to 8 and 1 to 6, is assumed by node 15, which links clusters 8 to 10 and 6 to 10. The loss of node 2 also results in the creation of the new gateway at 9, which links clusters 6 and 8. The loss of nodes 3 and 5 has no immediate effects.

Since the backbone networks shown in Frames (b) through (d) of Figs. 16 and 17 are the same, we only consider what happens in the case of the LCA (Fig. 15). The additional loss of nodes 11 through 15 has the effect of disconnecting the backbone network. This is unavoidable since, for example, the loss of node 14 results in the isolation of node 1. Likewise, the set of nodes 2, 8, and 9 are also isolated from the rest of the network once nodes 13 and 14 are lost. The cluster head roles of nodes 13 and 11 (Frame (b)) are taken over by nodes 6 and 10 (Frame (c)). The effects of the loss of head 15 are borne by new heads 6 and 10. That is, all the nodes within cluster 15 (Frame (b)) are now contained within the combination of new clusters 6 and 10 (Frame (c)). Also, the disappearance of 15 negates the need for a gateway node at 12. Thus the gateway role of 12 does not have to be taken over by any other node.

The additional loss of nodes 6 through 10 causes no further partitioning of the network; it still comprises three isolated parts. However, the roles of heads 6 and 10 and gateway 5 (Frame (c)) are now taken over by the single head at node 5 (Frame (d)). Again, we emphasize that, at other frequencies, the network may still be connected.

Loss of Links

So far, all the simulation examples shown have been for the case where: 1) the communication range is fixed in time, and 2) a single communication range is associated with each frequency. Condition (2) implies that all communication links are bidirectional. Both of our network structuring algorithms explicitly test whether links are bidirectional; consequently, these algorithms function properly even if condition (2) is not satisfied. In this section, we examine the effects that arise when condition (1) is not satisfied.

A time-varying communication range is likely to result in changing network connectivities. An important consideration is the rapidity of these changes with respect to the time it takes to organize the network (i.e. an epoch duration). If connectivities tend to remain constant over an epoch duration, then for most epochs our network structuring algorithms will work in the manner shown by our previous examples. Path redundancies, which occur as a result of the formation of distinct network structures during each epoch, can be used to advantage by avoiding the routing of traffic over any backbone networks that were improperly formed due to rapidly changing connectivities. Of greater concern is the performance of our network structuring algorithms when the connectivities change frequently with respect to the epoch duration. We give examples to show that even for the case of rapidly changing network connectivities our algorithms can provide reasonable network structures.

We begin by considering the network shown in Fig. 18a. The corresponding connectivity map, for a frequency of 10 MHz, is shown in Fig. 18b. The connectivities shown here are based on the groundwave range vs frequency curve of Fig. 8, which we have used in our previous examples. Applying the ALCA to a network with these connectivities, we obtain the network structure shown in Fig. 18c. The links shown here are the *primary* links of the network comprising the bidirectional links of the backbone network and the bidirectional links between each node and its own cluster head.

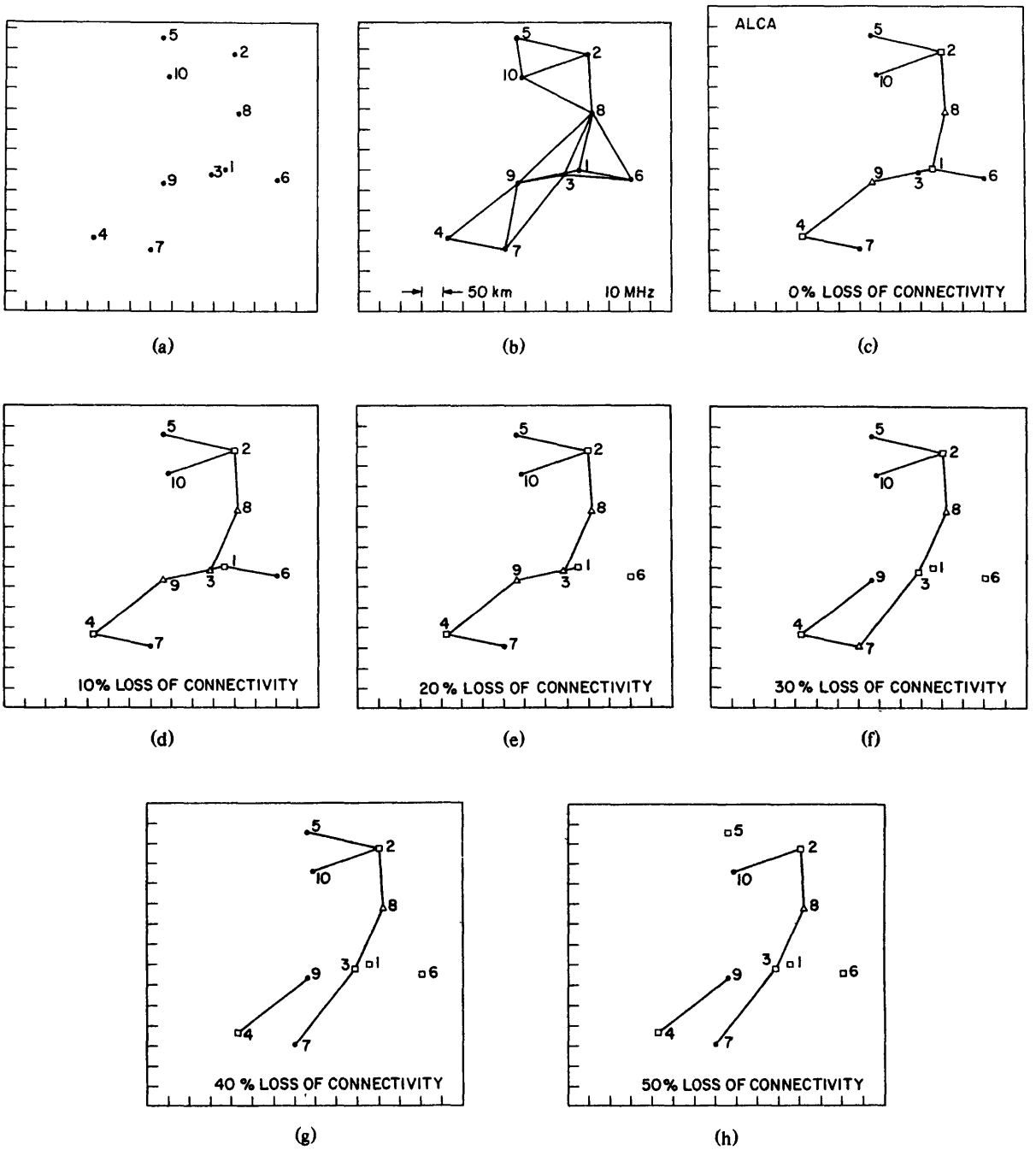

Fig 18 — This example illustrates the effects of a random loss of some one-way connectivities The distribution of nodes for this example is shown in (a) and the connectivity map is shown in (b). Frames (c) through (h) show the primary links that form under varying degrees of loss connectivity.

We repeat the simulation experiment, for cases of random pattern of connectivity losses as summarized in Table 2. (Because high-frequency channel noise can vary highly from one location to another, such a random pattern of connectivity losses is not unrealistic). We explain the results as follows.

Table 2 — Summary of link losses for the example shown in Figure 18

Percent loss of connectivity	Links Inoperative During Frame 1	Links Inoperative During Frame 2
10%	(8 to 1) (8 to 9)	(6 to 3) (6 to 8) (7 to 4)
20%	Above plus (6 to 1) (6 to 8) (9 to 3) (10 to 8)	Above plus (5 to 2) (8 to 9)
30%	Above plus (1 to 3) (3 to 6) (7 to 9) (10 to 5)	Above plus (1 to 9) (8 to 2)
40%	Above plus (6 to 3) (7 to 4)	Above plus (1 to 3) (4 to 7) (8 to 3) (9 to 7) (10 to 2) (10 to 5) (10 to 8)
50%	Above plus (3 to 9) (8 to 10) (9 to 1)	Above plus (2 to 5) (3 to 9)

For the example of a 10-percent loss of connectivity, as shown in Fig. 18d, the main effect is the creation of a gateway at node 3. This occurred because of the loss of the bidirectional link between nodes 8 and 1. Thus the former linkage of heads 1 and 2 using link (1,8) has been replaced by the linkage using links (1,3) and (3,8).

The network structure for a 20-percent connectivity loss is shown in Fig. 18e. There are two noticeable differences between this structure and the structure shown in Fig. 18d, namely, node 6 became a cluster head and 6 also became isolated from the rest of the network. Node 6 became a head because it is no longer bidirectionally connected to any lower numbered heads. Previously, it had been bidirectionally connected to node 1, but in the present case link (6 to 1) was lost. One might think that node 3 would link heads 1 and 6; however, since 3 did not "hear" 6 announce that it is a head (i.e. the link 6 to 3 was lost during Frame 2), the linkup was never performed.

The transition from a 20-percent to a 30-percent loss of connectivity results in: 1) the isolation of node 1; 2) node 3 now becomes a cluster head; 3) node 7 becomes a gateway node; and 4) node 9 is no longer a gateway. The isolation of cluster head 1 is caused by two factors. First of all, the bidirectional link to node 3 is broken due to the loss of link (1 to 3) during Frame 1. Second, node 9, which could link heads 1 and 4, is unaware that node 1 is a cluster head due to the missed Frame — 2 transmission from node 1. Consequently, node 9 does not become a gateway. Another result of the loss of link 1 to 3 in Frame 1 is that node 3 is left with no bidirectional links to lower numbered nodes. Thus node 3 becomes a cluster head. Node 7 now becomes a gateway for linking head 4 with the new head 3.

Proceeding to the case of a 40-percent loss of connectivity, the only change in the primary links from that of the previous case is the loss of gateway status for node 7, and consequently, the disconnection of heads 3 and 4. Node 7 does not become a gateway because it is no longer bidirectionally coupled to head 4 due to a missed (7 to 4) transmission during Frame 1.

For the case of 50-percent loss of connectivity, the net additional effect on the structure of the primary network is the isolation of node 5. This occurs because 5 erroneously assumes that it is not bidirectionally connected to 2 because of a missed transmission from node 2 during Frame 2.

The result of all this is that even when the connectivities are changing during an epoch, the network structuring algorithms can provide reasonable network structures. The previous example is also important because it gives some insight into the behavior of the algorithms when transmission errors occur since garbled transmissions can be modeled as time varying connectivities. Of course, a thorough quantitative analysis of the behavior of our network structuring algorithms in the presence of transmission errors and time varying connectivities must still be performed. However, such a study requires the consideration of other networking issues such as routing and channel allocation methods. Examples, such as those given in Fig. 18, should prove beneficial in guiding the selection of these routing and channel allocation strategies.

FURTHER CONSIDERATIONS

Our proposed network structuring algorithms only provide a method by which each node can determine whether it should assume the role of cluster head, gateway, or ordinary node. Still to be specified are the networking functions associated with each of these roles. Unfortunately, there still remain several important, unresolved networking issues that necessitate postponement of the detailing of these node functions. In this section we address some of these important issues. Although we do not propose definite solutions to the problems we raise, we do lay the groundwork for their eventual resolution.

Intracluster, Backbone, and Auxiliary Links

Intracluster Communication — The cluster structure is well suited to the use of certain multiple access protocols for communication between cluster members and their heads. Communication from a cluster head to the members of its cluster might be implemented either on a broadcast basis (i.e., a common transmission to all cluster members) or on a point-to-point basis. There are many possible schemes for communication from a cluster member to its head. They range from fixed assignment schemes (i.e., TDMA or FDMA) at one extreme, which are good for regular traffic, to pure contention schemes (such as ALOHA) which are good for bursty traffic. Other possible protocols include polling schemes, reservation schemes, and sensing schemes such as carrier sense multiple access (CSMA) and busy tone multiple access (BTMA). Many of these schemes and their potential applicability to the HF ITF network are discussed in Ref. 3.

Cluster heads will act as communication channel access controllers; the specific role of the cluster head would depend on the particular multiple access scheme used. In all cases the cluster head would forward the messages it receives toward their destinations (which may be either within the same cluster or external to it). Under TDMA, the head would have the reponsibility of establishing the frame structure after each execution of the LCA, as well as making changes to it if necessary. Under reservation schemes the cluster head would be responsible for allocating time slots or channels to platforms that request them. Under BTMA, [8] the cluster head would transmit a busy tone when one of its cluster members is transmitting to it, thereby instructing the other cluster members to remain silent. BTMA is especially well suited to bursty traffic in the cluster structure, since while all cluster members are within range of their head they are not necessarily within range of each other, therby creating the *hidden terminal* problem.

The implementation of multiple access schemes using spread spectrum signaling will differ somewhat from their narrowband counterparts. We shall consider some of the ramifications of using code division multiple access (CDMA) later in this section.

The Backbone Network — The cluster heads and gateway nodes and the links that connect them form what we call the *backbone network*. The backbone network is expected to carry a large portion of the intercluster communication load and to concentrate, relay and control that traffic through the network. Our network organizing algorithms allow for the simultaneous existence of several different backbone networks. However, constraints on the number of transmitters and receivers available at each

node are likely to play a major part in determining how many backbone networks are actually created and how they are used. Several factors argue against the use of random access protocols for communications over the backbone network. Since the nodes that form the backbone network are not generally all within one hop communication range of each other or of a central controller, it is not known how to solve the hidden terminal problem for these nodes. Furthermore, the traffic over the backbone network is likely to be less bursty than the intracluster case since each head acts like a concentrator for the intercluster traffic. This is so since most of the traffic from the nodes of one cluster directed to nodes outside the cluster is likely to exit the cluster only via the cluster head, and conversely most of the incoming traffic enters the cluster via the cluster head.

One possibility is to use circuit switched dedicated channels for the intercluster communication links. It will not be possible, due to signal interference (which will be discussed later) and equipment constraints to assign a circuit switched channel to every pair of nodes on the backbone network. However, dedicated links can be implemented via a TDMA process (that involves only a node and its neighbors, rather than all of the nodes in the ITF network).

Auxiliary Channels — The combination of intercluster (backbone network) links and intracluster links, which connect each node with its own cluster head, are referred to as the primary links of the HF ITF network. An illustration of the primary links for a sample network appear in Fig. 18c. The primary links of the HF ITF network might handle control traffic (for network management), signaling traffic (for call management), or voice/data traffic (i.e. user traffic). However, in many situations (e.g. when the communicating nodes are within direct range of each other) it may be preferable to set up auxiliary channels to handle some of the communication traffic (especially voice traffic and long data transfers). For example, with appropriate signaling, the network might set up separate dedicated circuits between pairs of ordinary nodes for the purpose of voice communications. The channels used to form these voice circuits might be entirely distinct from the channels used to form the primary links, and thus we call them auxiliary channels.

Routing

The proposed organizational architecture consists of node clusters, each with its own cluster head which is connected to other heads via the backbone network; there can be a set of these linked clusters corresponding to each of the M different epoch frequency ranges. Therefore, there may be many potential paths between a given pair of nodes. The network routing function must provide for the automatic selection of a suitable path for individual transmissions. We address here some of the features of the HF ITF network that relate to routing.

A special class of messages that may require several hops in their transmission (and thus need efficient routing) is the *functional net* traffic. In the task force environment, communications users are often grouped into functional nets. Members of the same functional net share a common objective and communicate with each other via the functional net. Each node may belong to several functional nets. There may be functional nets in support of antisubmarine warfare (ASW), antiair warfare (AAW), navigation/maneuvering, etc. These nets are usually either voice or data nets, but not both. A simple technique for establishing functional nets is to assign a dedicated frequency channel to each functional net. Although such a strategy is easy to implement, it is wasteful of communication resources which are often at a premium. Also, fixed frequency channels are easily jammed. The HF ITF network is more likely to use spread spectrum signals A number of routing issues arise in connection with the implementation of functional nets in a network that employs spread spectrum signaling and is structured into a linked cluster architecture; we briefly describe some of these.

Connectivity — Implicit in most discussions of routing is the concept of a direct link (channel) between a pair of nodes. It is via this link that information is sent from one node to another. For wire-line networks, it is usually obvious as to which nodes are directly connected. Hence, for these

nets, it is fairly straightforward to maintain a routing table which specifies expected delays along each path. Traffic can then be routed along the path of least delay. However, the situation is different in an HF radio network that employs frequency hopped (FH), spread spectrum, code division multiple access (CDMA) signaling. Let us consider the question of connectivity in this latter case. First, let us define actual and potential links. A link is said to be an *actual* link from node A to node B if node B is within communication range of A and if node B is monitoring the channel that node A uses to communicate with B. The link from node A to node B is said to be a *potential* link if node B is within communication range of A but is not monitoring the channel used by node A to communicate with B. By *channel* we mean a specific region in the time-frequency plane (for example, a frequency-hopped spread-spectrum code, or a TDMA slot, or a fixed, narrow frequency band). Nodes A and B are said to be *bidirectionally* connected if they are within communication range of each other.

We now return to our discussion of connectivity in an HF network which uses FH-CDMA signaling. If the signal is frequency hopped over a wide frequency range, say for example the entire HF band 2 to 30 MHz, a pair of nodes may be connected over only portions of the hopping sequence. Moreover, even at those frequencies where the nodes are connected, the connectivity may represent only potential and not actual links. Therefore, the connectivity information gathered during the execution of our network organizing algorithms is not, by itself, sufficient to allow us to proceed directly to the design of a routing strategy. First we must decide which links are to become active and how this information is to be made available throughout the network.

Communication with Nearby Nodes — The algorithms for deriving the linked cluster architecture provide each node with a list of its neighbors and its neighbors' neighbors at each frequency band. Thus it seems desirable that each node communicate directly with its neighbors and, when communicating with its neighbor's neighbor, to self-determine the appropriate relay node. Communication with nodes that are more than two hops away could proceed via the cluster head and the backbone network. In this manner, traffic over the backbone network would be reduced.

Broadcasts — The linked cluster architecture provides a convenient structure for broadcast-type transmissions. Here we use the term *broadcast* to mean that the transmissions are intended for many nodes. For example, a cluster head may wish to transmit a message to all members of its cluster, such as a busy tone whenever it is receiving a transmission from one of its cluster members. As another example, if a transmission is intended for all nodes in the network, it could be transmitted along the backbone network, and then each cluster head could broadcast the message to its own cluster members. The protocol for broadcast transmissions can be quite simple as long as acknowledgements are not required from the intended receiving nodes. The difficulty arises from the need to schedule acknowledgments from all recipients, as well as the need to rebroadcast the message if one or more of the intended destinations fail to receive correctly. Highly volatile missile-tracking information is an example of data that might be broadcast with no acknowledgment expected; transmissions to submarines that must maintain radio silence is another example. A functional network may be a broadcast network. However, all members of a functional network need not belong to the same cluster. In such cases, the message to be broadcast will have to be relayed throughout the functional network.

Spread Spectrum Signaling

The ITF network must use spread spectrum signaling techniques in order to provide some protection from jamming and interception of messages, as discussed in Refs. 3 and 19. The use of spread spectrum signaling leads naturally to the use of CDMA techniques, since under CDMA the dual purpose of providing multiple access capability as well as jam resistance can be achieved. We use the term CDMA to include all forms of spread spectrum multiple access, i.e., direct sequence (DS), frequency hopping (FH), and hybrid FH-DS signaling.

Under any CDMA technique the source transmits to the destination using a particular code. (For example, in the case of FH signaling the code corresponds to the FH pattern.) Division by code is analagous to division by time (as in TDMA) or frequency (as in FDMA). Under CDMA however, there is only quasi-orthogonality as opposed to full orthogonality among the codes of the different users. Therefore, signals transmitted using different codes can interfere with each other. (In the case of FH signaling, a *hit* occurs when two or more signals are transmitted simultaneously in the same frequency slot; forward error correction coding techniques can be used to handle the resultant loss of data, as long as the number of bits lost is not too great.) As the number of simultaneous transmissions (using different codes) increases, the interference level increases gradually, typically resulting in graceful degradation.

The basic principles of CDMA, and considerations relating to its use in the ITF network environment are outlined in Ref. 3. While it was tentatively concluded that hybrid FH-DS might be the best form of spread spectrum signaling for the ITF network, it is now believed that a pure FH system may be adequate and more practical based upon the complexity of the equipment required for a hybrid system [19]. In the discussion that follows, it is helpful to think of codes simply as FH patterns; the concepts may be extended easily to other forms of spread spectrum signaling. Owing to AJ considerations, the FH patterns must be of pseudorandom variety with extremely long periods, as opposed to periodic sequences, such as Reed-Solomon sequences, which have a period no greater than the number of frequency slots over which the signal is hopped [20].

The number of distinct (long period) pseudorandom FH patterns with good pairwise correlation properties is extremely large. A more relevant parameter is the number of signals taken from this set that can be transmitted simultaneously within communication range of each other without generating excessive interference. To get an idea of the numbers involved, we may consider a frequency bandwidth of 20 kHz, which results in 50 frequency cells per MHz. It is anticipated that typical network (subband) bandwidths would be between 2 and 5 MHz, resulting in between 100 and 250 frequency cells per subband. The signals of each of the M simultaneously operating networks would hop over the wideband channels, which consist of the full bandwidth of their respective subbands.

When a good set of FH patterns is used, the number of asynchronous FH signals that can simultaneously occupy a channel is approximately 1/10 of the number of frequency cells, although in practice the number would depend on the specific nature of the channel (e.g., the fading, dispersive, noisy characteristics of the HF channel), the modulation scheme, the relative strength of the interfering signals, the type of error correcting code that is used, and the tolerable bit error rates [19]. The multiuser problem is most severe in frequency bands that support substantial skywave propagation, because all platforms in the network can potentially produce interfering signals (rather than only those that are within groundwave propagation range), and because of multi-path propagation. It is clear that FDMA using narrow band signaling would permit a greater number of simultaneous active users than CDMA, since each of the frequency cells would be allocated to a separate signal with no chance of interference between signals. The use of spread spectrum signaling is necessary, however, to combat the jamming threat to the network.

We now discuss the application of CDMA techniques to the HF ITF linked cluster architecture. The basic code assignment considerations include the allocation of frequency hopping codes to users and the decision of whether to use a code associated with the transmitter or with the receiver. The four basic types of links in the network will be point-to-point, broadcast, random access, and common links. Considerations for these links differ, and so they will be considered individually.

Point-to-Point Links — When point-to-point links are used, the FH code of either the transmitting or receiving platform may be used, as long as consistency is maintained.

Broadcast Links — In a broadcast link, a single transmitter sends a common message to two or more platforms. It is therefore appropriate to use a FH pattern associated with the transmitting platform, and monitored by all potential receiving platforms.

Random Access Links — In a random access link many platforms attempt to communicate with a single platform on a contention basis. A FH pattern associated with the receiver is therefore essential, because the receiver will not know a priori which platforms are attempting to communicate with it.

Common Links — A common link, monitored by all network platforms, can be used for either broadcast or random access applications. A single FH pattern would be associated with such a link.

We may now discuss the number of FH codes required to implement the HF ITF Network. To simplify the discussion, we initially assume that only a single transmitter and a single receiver are available at each platform in every subband. We shall demonstrate that it is sufficient in this case to associate one transmitter FH code and one receiver FH code with each platform in the network, and that two additional networkwide codes are needed. The total number needed in a network of N platforms is therefore $2N + 2$. (This situation is of course repeated in each of the linked cluster networks simultaneously operating in other frequency bands.)

The links that must be supported in the linked cluster architecture are:

1) A channel, monitored by all platforms, to be used during the execution of the Linked Cluster Algorithm.

2) A random access channel that is monitored by all task force nodes.

3) A broadcast channel associated with each cluster head.

4) A random access channel associated with each cluster head.

5) Dedicated links for use on the backbone network.

6) Auxiliary links.

We now discuss the FH codes that are required for the implementation of these links.

1) The use of a single common code would be adequate for the execution of the linked cluster algorithm. It would not be necessary to use a unique code associated with each platform since the transmissions of each platform are inherently noninterfering due to the TDMA frame structure of the LCA.

2) The availability of a random access channel would allow a platform that is not part of the network formed by the LCA to communicate with a node that does belong to this network. This might occur, for example, when a rapidly moving aircraft or a previously radio-silent submarine wishes to communicate with a network node. Another example of such a need would be to accommodate the merger of two task forces. While equipment constraints would preclude the continuous monitoring of such a channel, a fraction of each receiver's time slots might be used for this purpose.

3) A broadcast code would be used by a cluster head to transmit simultaneously to all cluster members of its cluster, and each platform would have to monitor its cluster head's broadcast transmissions. In addition, since a node may be within range of several cluster heads, it might be worthwhile for it to monitor the broadcasts of other neighboring cluster heads in addition to its own; to do so would not require additional broadcast codes, but rather the ability to monitor additional codes. This would not necessarily require additional receivers, since it might be possible to monitor several cluster heads on a time division basis.

4) Intracluster communication between cluster members and their head can be implemented via either a contention-based mechanism or a contention-free mechanism. A receiver-based FH code is essential wherever contention is permitted. Either receiver-based or transmitter-based FH codes can be used in the contention-free case.

5) Backbone network links and auxiliary links are point-to-point links, and therefore can be implemented using the FH code of either the transmitter or receiver.

Under our (simplifying) assumption that only one transmitter and receiver are available on each platform in every subband, these equipment resources must be allocated among links with neighboring platforms on a time division basis. The code assignment problem is therefore transformed into a combined code and time slot assignment (channel allocation) problem. Some preliminary versions of distributed channel allocation schemes have been developed for the HF ITF network [21]. Such a time division also helps to limit to acceptable levels the number of signals that are simultaneously transmitted. Since each platform is limited to the simultaneous use of only one transmitter and one receiver, there would be no advantage gained by the assignment of a large number of codes to each platform.

We have shown that it is at least conceptually possible to implement a linked cluster network of N platforms, each of which has one transmitter and one receiver, by the use of $2N + 2$ frequency hopping codes.

It is straightforward to generalize the preceding discussion to the case in which each platform has several transmitters and receivers. It would be possible to associate a distinct code with each of them. Each platform would then be able to support additional communication links. It will be necessary to generalize the channel allocation algorithms to the case of multiple simultaneous channels per platform.

A remaining question is the procedure used to assign the frequency hopping patterns to the network platforms. The simplest approach is simply to associate a priori a pair of unique codes with each platform in the network. It is of course necessary for each platform to know the codes associated with the other network platforms. It may be possible to disseminate this information to all task force members before deployment of the task force. If not, then some mechanism will be needed to broadcast this information throughout the network. Security will be a major issue in the distribution of FH patterns, because it is essential that they are not known by an adversary.

Switching

A preliminary concept of the switching modes in the HF ITF network is the following. We expect that the primary communication links (i.e. the links of the backbone network plus the link between each node and its cluster head) are to be packet switched and used mainly for bursty data, control, and signaling traffic; voice traffic, long file transfers, and other nonbursty type traffic could use circuit switched auxiliary links. The auxiliary links will normally be set up by appropriate signaling over the primary communication network.

There are two reasons for splitting the traffic into bursty and nonbursty types. First, the nonbursty traffic (e.g voice or file transfers) may very well use up the entire communication capacity of a link of the primary network. This may seriously degrade the overall performance of the primary network. Second, because portions of the primary network may be reconfigured during each epoch of the network organizing algorithm, communication paths may need to be reconfigured. Such a reconfiguration of paths could cause unacceptable delays (or total disruption) of some of the voice traffic. On the other hand, auxiliary links need not change from epoch to epoch and thus auxiliary links seem preferable for voice transmissions.

Role of Aircraft and Submarines

In describing the organization of the network via the linked clusters, we have not addressed the special needs of receive-only nodes or of nodes that have rapidly changing network connectivities Our cluster forming algorithms tacitly assume that the network connectivities are relatively (with respect to the epoch duration) slowly varying and that all nodes can both transmit as well as receive. The first assumption may be violated by some fast moving aircraft while the second assumption may be inconsistent with the need for submarines to maintain radio silence for extended lengths of time. In particular, standardized protocols that require acknowledgments will not be suited for use by submarines. Therefore submarines and some nonstation keeping, fast moving aircraft are considered to be special nodes that do not participate in the formation of the backbone network. Their integration into the ITF network is yet to be specified.

CONCLUSIONS

In this report we have considered the design of an architecture for the HF ITF network. We have proposed and argued for an architecture consisting of node clusters, locally controlled by cluster head nodes, and linked to each other via gateway nodes. We have obtained two different algorithms for achieving this architecture. Both algorithms can be implemented in a distributed fashion and can adapt to topological changes. Thus they allow the Task Force to self-organize and achieve the suggested architecture from arbitrary initial conditions and to trace and gradually adapt to arbitrary topological changes. The properties of the HF channel are explicitly taken into account and used to advantage by both algorithms. These algorithms have been exhaustively studied and simulated. They have been shown to work properly and in a manner that satisfies many of the requirements of the HF ITF network design. The overall design for the full operation of the network requires the specification of many functions and parameters that transcend the scope of this report. However, we have given preliminary exposure to many of these functions and shown how the proposed architecture provides a suitable design framework for them.

REFERENCES

1. D. Barrick, "Theory of HF and VHF Propagation Across the Rough Sea, 2, Application to HF and VHF Propagation Above the Sea," Radio Science, Vol. 6, No. 5, pp. 527-533, May 1971.

2. CCIR Report 322, "Worldwide Distribution and Characteristics of Atmospheric Radio Noise," International Radio Consultative Committee, Documents of the Xth Plenary Assembly, Geneva 1963.

3. J.E. Wieselthier, D.J. Baker, and A. Ephremides, "Survey of Problems in the Design of an HF Intra-Task Force Communication Network," NRL Report 8501, Oct. 1981.

4. R.E. Kahn, et al., "Advances in Packet Radio Technology," Proceedings of the IEEE, Vol. 66, No. 11, Nov. 1978.

5. N. Ehrlich, "The Advanced Mobile Phone Service," IEEE Communications Magazine, Vol. 17, No. 2, pp. 9-15, Mar. 1979.

6. A. Nilsson, W. Chou, and C. J. Graff, "A Packet Radio Communication System Architecture in a Mixed Traffic and Dynamic Environment," Proceedings of the Computer Networking Symposium, IEEE Catalog No. CH1586-7/80, pp. 51-66, 1980.

7. M. Lawson, and S. Smith, "Reconfiguration Techniques of a Mobile Network," Proceedings of the 1980 International Zurich Seminar on Digital Communications, IEEE Catalog No. 80CH1521-4 COM, pp. B10.1 - B10.5, 1980.

8. 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 Transactions on Communications, Vol. COM-23, No. 12, pp. 1417-1433, 1975.

9. D.J. Baker and A. Ephremides, "The Architectural Organization of a Mobile Radio Network via a Distributed Algorithm," IEEE Transactions on Communications, Vol. COM-29, No.11, pp.1694-1701, Nov. 1981.

10. Y.K. Dalal, "Broadcast Protocols in Packet Switched Computer Networks", Stanford University, Dept. of Electrical Engineering, Technical Report No. 128, Apr. 1977.

11. R.G. Gallager, P.A. Humblet, and P.M. Spira, "A Distributed Algorithm for Minimum Weight Spanning Trees," Massachusetts Institute of Technology, Laboratory for Information and Decision Systems, Technical Report No. LIDS-P-906-A, Oct. 1979.

12. A. Hauptschein, and M. Kajor, "Recognition and Self- Organization of Nodes into DTDMA Nets," IEEE Transactions on Aerospace and Electronic Systems, Vol. AES-17, No.4, pp. 531-542, July 1981.

13. D.J. Baker, and A. Ephremides, "A Distributed Algorithm for Organizing Mobile Radio Telecommunication Networks," Proceedings of the Second International Conference on Distributed Computing Systems, pp. 476-483, Apr. 1981.

14. G. Birtwistle, O.J. Dahl, G. Myrhaug, and K. Nygaard, SIMULA begin, Auerbach Publishers, Philadelphia, 1973.

15. C.E. Landwehr, "Satellite Communications Simulator - Concepts and Facilities," NRL Tech. Memo. 7503-111, 6 Apr. 1978.

16. C.E. Landwehr, "SIMULA and Events," NRL Tech. Memo. 7503-113, 11 Apr. 1978.

17. C.E. Landwehr, "An Abstract Type for Statistics Collection in SIMULA," ACM Transactions on Programming Languages and Systems, Vol. 2, No. 4, pp. 544-563, Oct. 1980.

18. "Naval Material Command Naval Telecommunication Development Plan, Vol.II Design Guidance (U)," Western Electric Company, 31 Jan. 1977 (Secret).

19. J.E. Wieselthier, D.J. Baker, A. Ephremides, and D.N. McGregor, "Preliminary System Concept for an HF Intra Task Force Communication Network," to be published as an NRL report.

20. D.V. Sarwate, and M.B. Pursley, "New Results on Frequency-Hopping Patterns for Spread-Spectrum Multiple Access Communications," Proceedings of the International Conference on Communications, 1980.

21. D.J. Baker, and A. Ephremides, "Distributed Assignment of Links in Mobile Radio Networks," paper presented at the Nineteenth Annual Allerton Conference on Communication, Control, and Computing, 1981.

Appendix
PSEUDOCODE FOR THE NETWORK ORGANIZING ALGORITHMS

In this appendix we present SIMULA-based pseudocode of the linked cluster algorithm (LCA) and the alternative linked cluster algorithm (ALCA) for node *i*. We also give pseudocode for procedures LINKUP1 and LINKUP2.

Pseudocode for the LCA

```
COMMENT: Initialize data structures;
OWNHEAD:=SELF;
NODESTATUS:=HEAD;
HEADSONEHOPAWAY:=HEADSTWOHOPSAWAY:=NODESHEARD:=EMPTY;
CONNECTIVITY:=IDENTITYMATRIX;
Schedule LCA transmissions at this node;
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: Received node j's transmission;
                             Put node j into NODESHEARD list;
                             IF node i heard by node j, CONNECTIVITY[i,j]:=1;
            COMMENT: Frame 2 events;
            Transmit: Determine NODESTATUS;
                             Broadcast CONNECTIVITY vector and NODESTATUS;
            Receive: COMMENT: Received node j's transmission;
                             Fill in row j of CONNECTIVITY matrix;
                             IF j<i, CONNECTIVITY[i,j]:=CONNECTIVITY[j,i];
                             IF node j = HEAD, put node j into HEADSONEHOPAWAY;
    END;
COMMENT: Fill in HEADSTWOHOPSAWAY list;
For each node m that is bidirectionally connected 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 HEADSTWOHOPSAWAY;
    END;
COMMENT: Call procedures to perform cluster linkage;
DELETEHEADS;
LINKUP1;
LINKUP2;
```

Pseudocode for the ALCA

```
COMMENT: Initialize data structures;
OWNHEAD:=SELF;
NODESTATUS:=HEAD;
HEADSONEHOPAWAY:=HEADSTWOHOPSAWAY:=NODESHEARD:=EMPTY;
CONNECTIVITY:=IDENTITYMATRIX;
Schedule ALCA transmissions at this node;
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: Received node j's transmission;
                             Put node j into NODESHEARD list;
                             IF node i heard by node j, CONNECTIVITY[i,j]:=1;
            COMMENT: Frame 2 events;
            Transmit: Determine OWNHEAD;
                             Broadcast CONNECTIVITY vector and OWNHEAD;
            Receive: COMMENT: Received node j's transmission;
                             Fill in row j of CONNECTIVITY matrix;
                             IF j<i, CONNECTIVITY[i,j]:=CONNECTIVITY[j,i];
                             IF head of node j is node i, do nothing
                             ELSE IF head of node j is neighbor of node i THEN
                             Put head of node j into HEADSONEHOPAWAY
                             ELSE put head of node j into HEADSTWOHOPSAWAY;
    END;
COMMENT: Call procedures to perform cluster linkage;
LINKUP1;
LINKUP2;
```

Pseudocode for the Procedure LINKUP1.

```
If node i is not a cluster head then
BEGIN
    COMMENT: Node i is a candidate to be a gateway;
    For all combinations of pairs of heads 1-hop away from node i do
        BEGIN
            If heads under examination are not directly connected
            nor connected via another head then
            BEGIN
                Rank all the potential gateways for the examined pair of heads;
                If node i is the highest ranking candidate then node i
                becomes the gateway for the examined pair of heads;
            END;
        END;
END;
```

Pseudocode for Procedure LINKUP2.

```
If node i is not a cluster head then

BEGIN
    COMMENT: Node i is a candidate to be a gateway node;
    For all pairs of heads (h1,h2) consisting of node i's own head
    h1 and head h2, which is 2-hops from node i do
        COMMENT: Check for prior linkage of h1 and h2;
        If no neighbor of node i is connected to both h2 and one of i's
        1-hop away heads then
        BEGIN
            Rank all the potential gateway-to-gateway links, which join a
            node from cluster h1 to a node from cluster h2;
            If node i is an endpoint node on the highest ranking link then
            node i becomes a gateway for linking h1 and h2;
        END;
END;
```