Multicast Routing in Internetworks Using Dynamic Core Based Trees

A.D. Raghavendra and S. Rai
Department of Electrical and Computer Engineering
Louisiana State University, Baton Rouge

S.S. Iyengar
Department of Computer Science
Louisiana State University, Baton Rouge

ABSTRACT As the network size increases, traditional approaches to multicasting such as distance-vector and shortest-path-first routing demand more bandwidth, memory, and processing resources at each router. A recently proposed architecture, called Core Based Tree (CBT) [1], overcomes these problems by building only one tree per multicast group. The CBT approach, however, does not route packets through the shortest possible path. This paper presents a modification to the CBT tree building algorithm so that minimal spanning trees are built when needed and packets are routed through nearly optimal paths. We also give a procedure to choose the core of the tree thus built and this, in turn, helps reduce the multicast delay. Examples, using SURAnet topology, are solved to illustrate our approach.

1. Introduction

The multicast facility is becoming increasingly popular in computer networks because it offers services like audio and video conferencing [11], resource discovery, replicated database updating and querying, software update distribution, and stock market information [12]. Multicasting is the transmission of a message to a group of computers identified by a single destination address. It can be considered to be a generalized form of broadcasting and unicasting. Multicast transmission reduces the overhead on the sender - it is cheaper than broadcasting to all members or unicasting to each member in a group. It also reduces the load on the network and the time taken to reach all destinations.

Two common routing algorithms used to provide conventional multicast are distance-vector and link-state algorithms. In the distance vector algorithm each participating node periodically sends out lists of destinations it can reach to and the distances to them (at startup, each node has a list of destinations to which it is directly attached). With link-state routing, each node has complete topology information about the network and computes routes independently using Dijkstra's algorithm [2]. Nodes test their links periodically and broadcast that information. Reference [6] surveys these techniques and also the single-spanning-tree algorithm. It, further, discusses the way these algorithms are combined to provide multicast in large internetworks. Chow in [3] proposes an optimal algorithm and two polynomial time multicast heuristics that compute nearly optimal paths. As the network size increases, these methods demand more bandwidth, memory, and processing resources at each router or node. Recently, an architecture, called *Core Based Tree* (CBT), is proposed in [1] to overcome these problems. The *Simple Internet Protocol Plus* (SIPP) is a next generation *Internet Protocol* (IP) and accomodates both conventional and CBT multicast techniques [7].

For communication, send an e-mail to suresh@ee.lsu.edu or call Dr. Rai at (504) 388-4832.

The CBT system of multicasting builds one tree for a multicast group. The same tree is used to provide multicast to the group for all sources. One router in the tree is chosen to be the *core*. The other routers are called *noncore* routers. The core's address is available to any sender who wishes to multicast to the group. When a packet needs to be multicast to the group, the sender sends the packet to the core node by unicast. If the packet hits any router of the group before reaching the core node, it is multicast to the group (the multicast address of the group is put in the destination field of the packet). This tree may not route multicast packets through the shortest possible paths. This paper presents a modification to the CBT tree building algorithm so that the CBT tree (minimal spanning trees comprising group members) is rebuilt whenever a large number of nodes have joined/exited the group. As a result, packets are routed through nearly optimal paths. We also give a core (or center) selection procedure for the group tree. We choose the core of the tree such that the multicast delay to the group is further reduced.

This paper is organized as follows: We review conventional multicasting techniques in Section 2. Section 3 discusses the CBT system of multicasting. It then lists the merits and demerits of CBT architecture. It also describes the way we have addressed problems in our proposed algorithm. Section 4 discusses dynamic core based trees. It describes tree building and core selection methodologies. It also provides the algorithms that run on core and noncore routers. A simulation of the proposed algorithm on SURAnet topology is studied as an example. Section 5 discusses implementation details and provides the timing analysis of the algorithm. Finally, Section 6 summarizes the paper.

2. Multicasting Techniques - An Overview

2.1 Preliminaries

For details on an IP multicast model, refer to [5]. In a multicast group, the transmitter need not know the individual identities of the group members. It sends the message to the destination address given to the group. The members need not be on the same network. Membership in a group is dynamic. A host may join or leave a group at any time. There is no restriction on the number of members in a group. Also, a host may be a member of more than one group at a time. A host need not be a member of the group to send messages to it. IP multicast groups may be permanent or temporary. The permanent groups have a well-known address assigned to them and exist at all times. Other multicast addresses are used by temporary groups which exist only when they have members.

In case members of a group are located on more than one network, the multicast routers of the network (which receive all multicast messages) forward the message toward the destination(s). The time-to-live (TTL) field in a message limits its

0-7803-3255-5/96 $5.00 © 1996 IEEE

propagation, just like the TTL field in a unicast message limits its propagation through an internet.

To make inter-network multicast possible, a host needs to inform the local multicast router(s). It must inform the router(s) about its group membership(s). The local router then broadcasts this information to other routers that can use the information to build multicast tree(s) for the group(s). Routers use the *Internet Group Management Protocol* (IGMP) to propagate group membership information.

IGMP has two phases. In phase 1, when a host joins a group it sends a message to the local multicast router(s) which, then forward the information to other multicast routers. Phase 2 deals with group updating. Because group membership is dynamic, multicast routers periodically broadcast poll messages to find out which hosts are members of which groups. They, then, broadcast this information to other routers.

2.2 Conventional Techniques

When routers exchange routing information, they use one of the two basic methods, distance-vector or shortest-path-first scheme. In what follows we discuss these techniques that are used to distribute routing information and build multicast trees.

The name distance-vector comes from the information in the messages that are sent out periodically. A message contains a list of pairs (V,D), where V is the destination and D is the distance to the destination. Each node begins with a set of routes to nodes to which it attaches. It keeps a list of routes in a table, where each entry identifies V and gives the distance to that node, measured in hops. Periodically, each node sends a copy of its routing table to any node it can reach directly. The receiving node updates its table accordingly. As an example, consider a report arriving a node K from node J. Node K examines the list of destinations and the distances to each. If J knows a shorter way to reach a destination, or if J lists a destination that K does not have in its table, or if K uses J to reach a destination and J's distance to that destination changes, K replaces its table entry.

The distance-vector algorithm is implemented for IP multicast and is called *Distance Vector Multicast Routing Protocol* (DVMRP). In this algorithm, for each multicast group, the nodes build a tree based on routing information (that is, the nodes build a routing tree on top of the graph of physical connections). When a node receives a message for a group, it determines which of its links relative to the source are in the shortest path tree. It, then, sends out a copy of that message on the links that correspond to branches in the routing tree. The process of excluding branches not in the shortest path tree is called *pruning*. A network is called a *leaf* network, for a particular source, if no other node uses that network to reach the source. If such a network has no members of a particular multicast group, then this network can be *truncated* or *pruned* from the shortest path tree for that group.

The other type of algorithm used to build multicast trees is called link-state or shortest-path-first (SPF) algorithm. Here, each participating node must have complete topology information. Alternatively, each node has a map that shows all the nodes and the edges in the network. The protocol performs two tasks. First, it tests the status of all neighbor nodes. Second, it sends out this link status information to all the other nodes. Refer to [4] for details of this protocol. Whenever a link status message arrives, a node updates its *map* of the graph. In case of any change, it recomputes routes by applying Dijkstra's algorithm [2]. Dijkstra's algorithm computes the shortest path to all destinations from a single source.

3. CBT Multicast

In this system of multicasting a single tree is formed which comprises members of a multicast group [1]. One of the routers (nodes of the tree) is chosen to be the core. Additional cores may be chosen for reliability. Reference [1] provides the details of the CBT architecture and operation of the protocol. In the following, we consider multicast group and operation components of this approach.

Multicast groups: A multicast group is identified by a group ID, similar to an IP multicast address (class D address). A directory service is maintained which contains the names, corresponding group-ids, and core addresses of all the multicast groups currently present. When queried with a group name the service answers with the corresponding group-id and core address for the group.

Operation: A host which wishes to multicast to a group sends the multicast packet to the core of that group (the core address can be obtained from the directory service). It puts the IP address of the core of the multicast group in the destination address field of the packet. It then uses normal unicast routing to send the packet to the core. Once the packet reaches any router on that multicast tree it is multicast throughout the tree. i.e. on arrival at an on-tree router the core address in the destination field of the packet is replaced by the multicast group address. Thus, it need not actually reach the core to be multicast to all the tree routers. This reduces packet switching and load on the network.

Conventional multicasting techniques, discussed in Section 2, are source-based as there exists one tree for each source in the multicast group. If S is the number of sources and N is the number of groups, a source based technique requires to store SN units of information in each router. A CBT technique, on the other hand, is a receiver-based approach. It forms only one tree per multicast group and, thus, has to store only N units of information in each router. This results in much better performance when a large number of multicast groups are in existence at any one time.

Only routers belonging to a multicast tree are required to store information about the tree and do preprocessing for the tree in a CBT system. The DVMRP, an existing algorithm, requires routers to do so although they form no part of the tree [14].

Existing multicast algorithms, discussed in Section 2, assume that all systems run the same multicast/unicast algorithms. Note that the Internet comprises a large number of heterogeneous systems and, therefore, the assumption is stringent or difficult to fulfill. A CBT protocol requires information from the router's forwarding (routing) table but does not require any knowledge of the router's unicast algorithm. Thus, it can be used in a heterogeneous environment. Developers need not consider multicasting while modifying/changing underlying unicast methods [1].

Core based trees may not give the shortest path between nodes of the tree. This problem is addressed in the algorithm presented in this paper. See subsection 4.3 for an example. Failure of the core results in the cessation of multicasting for the group. This problem is solved by having multiple cores for a tree. Upon failure of the primary core, one of the other cores take over, based on a previously assigned priority. Refer to [1] for failure recovery options.

4. Dynamic Core Based Trees

The algorithm, given in subsection 4.1, builds core based minimal spanning trees (MSTs) for the graph comprising routers of the multicast group. Note that for creating MSTs, a cost is involved each time a router is added (deleted) to (from) the tree. Therefore, we choose a dynamicity parameter, α, for the tree and the MST is built whenever α exceeds a prescribed threshold value α_{in} (generally taken as 0.1 for achieving better performance). The parameter α is defined as the ratio of the change in the number of nodes to the total number of nodes present in the multicast group. At each router of the tree (i.e. at each node of the graph) the algorithm in subsection 4.1 is used to forward packets. Each router determines which of its links are part of the MST. Then, it forwards copies of the packet on all such links except the one it arrived on.

We have addressed the problem of dynamic core placement too. An algorithm that builds an MST comprising members of a multicast group and (at each node) broadcasts packets onto all MST links (connected to the node) except the one it arrived on generates the absolute minimum number of packets to do the job [13, p. 308]. Thus, this scheme is optimal with respect to packet switching overhead and bandwidth consumption irrespective of the core placement. However, it is non-optimal if delay is considered. The core selection procedure is explained below.

The tree has one or two centers depending on whether the diameter of the tree has an odd or even number of nodes. The diameter of the tree is defined to be the longest path in the tree. Also, the diameter has to be on a path starting on a node of degree one. Refer to [9] for a detailed discussion on finding the center of a tree. Selecting a core or cores each time the network topology changes is expensive. Therefore, a core selection parameter (β) that takes into consideration the group dynamicity is chosen. We define parameter β as the ratio of the number of nodes added/deleted (since the core was last chosen) to the total number of nodes in the multicast group. Depending on the value of β, we execute the core selection procedure. The core is re-selected depending on how much the group topology has changed since the core was last selected. Based on the percentage change in the group topology, the following actions are taken.

Case 1: No action is taken if the change is below 3 percent (β<0.03). Note that the change is too small to justify the overhead of reselecting the core.

Case 2: The change is said to be significant if it is between 3 percent and 10 percent. The change is not enough to warrant the execution of the core selection procedure. In this case, the core is moved one hop towards or away from the locality of the change (depending on whether nodes are added to or deleted from the group). If nodes join the group, the core is moved, one hop, towards them. If nodes exit from a group, then the core is moved, one hop, on a path not leading to the exiting nodes. This reduces the distance from the core to the node farthest away from it. The distance from the core to the node at the greatest distance from it is minimized if the newly selected core is the center of the tree. Moving the core, one hop, is much faster than executing the core selection procedure (using Dijkstra's algorithm), and, when the core selected happens to be the center of the tree, this has the same effect as executing the core selection procedure (see Case 3).

Case 3: A considerable change is present if β is greater than 10 percent (β>0.10). A reselection of the core is likely to reduce the distance from the core to the node farthest away from it. As a result, the maximum delay to provide multicast to the group might be reduced. The center of the tree is found as follows: Dijkstra's algorithm is executed simultaneously on all nodes of degree one. Any of the longest paths thus found is a diameter [9]. The center of this path is chosen to be the core. In case the diameter has an even number of nodes, there are two centers, each at a distance of one hop from the other. The node with the lower IP address (a unique 32 bit number for a machine) is chosen to be the core (although any one of them can be chosen to be the core).

4.1 The Algorithm

The algorithm, in pidgin ALGOL (see [10]), is given below.

```
Core Router:
/*It computers α, builds core, and transmits messages.*/
        begin
        read α_in;
        while TRUE do
            begin
                COMPUTE_ALPHA;
                if α_in < α
                    begin
                        BUILT_MST;
                        CHOOSE_CORE;
                    end;
                SEND_MSG;
            end
        end
Non-Core Router:
/*The code for the router executes procedure SEND_MSG whenever there is a message to forward. The router forwards the incoming message out over all links which are part of the MST for the multicast group (except the one over which the message arrived).*/
        begin
        while TRUE do
            SEND_MSG;
        end
/* Let N_e be the number of nodes currently present in the tree, δ be the current change in the number of nodes and Δ be the change in the number of nodes since the core is chosen. */
procedure COMPUTE_ALPHA:
        begin
            α = δ/(N_e + δ)
        end

procedure CHOOSE_CORE:
        begin
            COMPUTE_BETA;
            if β > 0.03 and β <=0.10 then
                begin
                    move core, one hop, towards/away
                    from node(s) joining/exiting group;
                end
            else
                if β > 0.10 then
                    begin
                        execute Dijkstra's algorithm on
```

```
                    nodes with degree one;
                    choose center of diameter as core;
                    /* in case the diameter has an even
                    number of nodes, choose center
                    with lower id */
                end
    end
procedure SEND_MSG:
    begin
        copy Msg from input buffer;
        send Msg on all edges connected to node, which are
        part of MST, except the one it came on;
    end
procedure BUILD_MST:
    At each node:
    begin
    /*Begin with a group of fragments (which are nodes
    themselves initially)*/
    Each fragment finds the minimum weight outgoing
    edge, adds itself to it and passes on this information to
    the fragment at the other end of the link;
    end
procedure COMPUTE_BETA
    begin
        β = Δ/(Ne + Δ)
    end
```

The distributed algorithm for building MSTs is faster than the Prim-Dijkstra algorithm for building MSTs [8]. It also avoids producing cycles in case multiple MSTs exist. It happens because BUILD_MST uses node identifiers that are unique IP addresses.

Lemma. The procedure BUILD_MST provides a loop-free path.

Proof: The unique IP address helps select right link to avoid loop formation. Here, in the case of equal weight links, preference is given to the link with the lower identity end node, and, if these are same, tie is broken in the favor of a link whose other node has the lower identity. This observation, given in [2, p. 315], helps prove the Lemma. ■

4.2 Discussion

The parameter, α, can be chosen taking into account the group's characteristics. A desired property in a multicast protocol is flexibility because multicast applications have diverse characteristics with factors like number of senders, group size, traffic, and membership dynamicity dictating group characteristics.

The packet forwarding procedure for the algorithm, described earlier, in which a copy of the packet is forwarded on to all spanning tree (corresponding to the multicast group) links except the one it arrived on, results in the least possible number of packets begin generated. This results in minimum packet switching overhead and bandwidth consumption for the network.

There are two concepts for the center of a tree [9]. One is based on minimizing the largest number of hops and the other is based on minimizing the average number of hops. In the first case, the maximal node to node distance in the network is minimized. In the second case, the average node to node distance in the network is minimized. As described earlier, our algorithm sends a single packet over a link which might be shortest path to more than one node. Only when the shortest path diverges to lead to the individual nodes, copies of the packet are generated and sent on those diverging paths. If one packet was generated for each destination (at the source itself), then minimizing the average number of hops would result in least overhead (packet switching and bandwidth consumption). In our case, packet switching and bandwidth consumption are already minimum (refer to Section 4). The parameter to be considered is delay. Delay is determined by how far a node is away from the center. Choosing the center of the diameter as the center of the tree minimizes the greatest distance (and consequently the greatest delay) from the center to any node in the tree. By finding the center of the diameter, and choosing that as the core (center), we are minimizing the largest delay. So, the first approach of finding the center is chosen in subsection 4.1.

4.3 An Example

A simulation of the algorithm is performed using SURAnet topology. A set of DEC 5000/120 workstations were used and the code was written in C using the Berkeley socket library. Figure 1 shows the SURAnet map with all the nodes and links. The algorithm took two iterations to build the MST. Figure 2 shows the state after completion of the first iteration. Figure 3 shows the MST obtained after the second iteration.

Now, two cases are shown - one in which α is less than α_{in} and the other in which α is greater than α_{in}. α_{in} is assumed to be 0.10. In figure 4, node MSB decides to leave the group resulting in an α equal to 0.06 (1/18). Note that α (0.06) is still less than α_{in} (0.10). So, no new MST is built and the existing MST is retained. Figure 5 shows the state after node MSB has left the group. For case 2, we consider figure 6 that illustrates a multicast group. Assume the nodes WVnet, SURA, WTN, NOF and CTV decide to join the group, one after the other. The resultant CBT tree is shown in figure 7. Here α is 0.26 (5/19), and is greater than α_{in} (0.10). This starts the MST building process and the result is the tree shown in figure 8. The network shown in figure 7 (CBT tree) has a diameter of 16 (hops) whereas the network in figure 8 (the tree obtained after the algorithm in subsection 4.1 is run) has a diameter of 12 (hops). In this case running the algorithm given in subsection 4.1 resulted in a 25 percent decrease in the diameter of the tree. This means that the maximum delay for multicast packets to reach all nodes would be reduced by 25 percent.

5. Implementation Issues and Complexity Analysis

We, now, present the implementation details of the server and the time complexity of the method given in subsection 4.1.

5.1 Implementation Issues

Procedure SEND_MSG uses a concurrent TCP/IP server. TCP is chosen because the transport protocol takes care of packet loss and out-of-order delivery problems automatically; the program need not bother about them. In addition it provides reliability - it informs the process in case of a connection failure and it makes sure that the data arriving is error-free [4].

A concurrent implementation is chosen because the router may receive requests faster than it can process them. The program runs forever, waiting for connection requests to arrive. When a connection request arrives at a designated input port (which is the same port number on all machines running this protocol - port 7000 in our simulation), the program forks a process to service the requested multicast. Then the parent process continues to wait for an incoming connection. The maximum length of the connection request queue in our implementation is five. i.e. it

can enqueue up to five connection requests while the program is busy forking off a process to handle the request.

The child process handls the request. It opens connections to all the nodes of the MST except to the node on which the MSG has arrived. i.e. it opens TCP connections (see function connect TCP) to the nodes. Next, it reads from its designated input port. Then it writes MSG to all sockets (TCP connections to nodes).

5.2 Complexity Analysis

Reading in α takes a constant amount of time. Also, COMPUTE_ALPHA and COMPUTE_BETA take a constant amount of time since all the steps in these procedures are program statements not involving graph parameters. Note that CHOOSE_CORE includes Dijkstra's algorithm and a step that computes the core. The core computation step takes a constant amount of time while the complexity of Dijkstra's algorithm is $O(n^2)$. Here, n is the number of nodes in the network.

Procedure SEND_MSG has a complexity of $O(n)$ since in the worst case this procedure may operate forwarding messages serially. It will then take $n-1$ hops for a message to reach to a farthest node.

The distributed MST building algorithm has a worst-case complexity of $O(n\log_2 n)$ [8].

Procedure CHOOSE_CORE determines the time complexity of the algorithm. So, our algorithm requires a time complexity of $O(n^2)$.

6. Summary

The CBT system overcomes the conventional multicasting problems associated with large networks. However, the trees built by the CBT system may not provide a shortest path between nodes in the tree. The proposed technique builds minimal spanning trees based on how fast the network topology changes. We have used a parameter α to measure the network dynamicity. It is contrary to having a static CBT tree structure permanently servicing multicast requests for a group. Also, a procedure for choosing the core (center) of the tree is described. A core refers to the node to which all multicast requests are sent. Placement of this core is critical for multicast delay in the group. Choosing a core each time the network topology changes results in a lot of overhead. Therefore, our technique has used a parameter, β, that helps select the core based on network dynamics. We have verified our approach based on a simulation using SURAnet topology.

A full scale implementation on the actual TCP/IP internet would give a better idea on what values to choose for α and β for an application. Also, other issues such as dynamic group-id assignment and security have to be incorporated into the algorithm.

References

[1] T. Ballardie, P. Francis, and J. Crowcroft, "Core based trees (CBT)", *ACM SIGCOMM'93*, Ithaca, NY, Sept. 1993, pp. 85-95.

[2] D. Bertsekas, R. Gallager, *Data Networks*, Prentice-Hall, Englewood Cliffs, New Jersey, 1987.

[3] C. Chow, "On multicast path finding algorithms", *Infocom'91*, April 1991, pp. 1274-1283.

[4] D.E. Comer, *Internetworking with TCP/IP vol. I: Principles, Protocols and Architecture*, 2nd Ed., Prentice-Hall, Englewood Cliffs, New Jersey, 1991.

[5] S.E. Deering, "RFC 1112, Host extensions for IP multicasting", SRI Network Information Center, 1990.

[6] S.E. Deering and D.R. Cheriton, "Multicast routing in datagram internetworks and extended LANs", *ACM Transactions on Computer Systems*, vol. 8, no. 2, May 1990, pp. 85-110.

[7] P. Francis and R. Govindan, "Flexible routing and addressing for a next generation IP", *ACM SIGCOMM'94*.

[8] R.G. Gallager, P.A. Humblet, and P.M. Spira, "A distributed algorithm for minimum-weight spanning trees", *ACM Transactions on Programming Languages and Systems*, vol. 5, no. 1, January 1983, pp. 66-77.

[9] P. Hamalainen, *Algorithms for some center problems on networks*, Jyvaskylan Yliopissto, Jyvaskyla, 1984.

[10] J.E. Hopcroft, A.V. Aho, and J.D. Ullman, *Data Structures and Algorithms*, Addison-Wesley, Reading, MA, 1983.

[11] "Report: multicasting from the second international world web conference 1994: mosaic and the web", Chicago, 17-20 October 1994.

[12] "Streaming (tm) multicast LAN technology", This artical can be viewed on the world wide web at address: http://www.intel.com/comm-net/cnn_work/cn_multi.html

[13] A. Tanenbaum, *Computer Networks*, Prentice-Hall, Englewood Cliffs, New Jersey, 1981.

[14] D. Waitzman, C. Partridge, and S. Deering, "RFC 1075, distance vector multicast routing protocol", SRI Network Information Center, 1990.

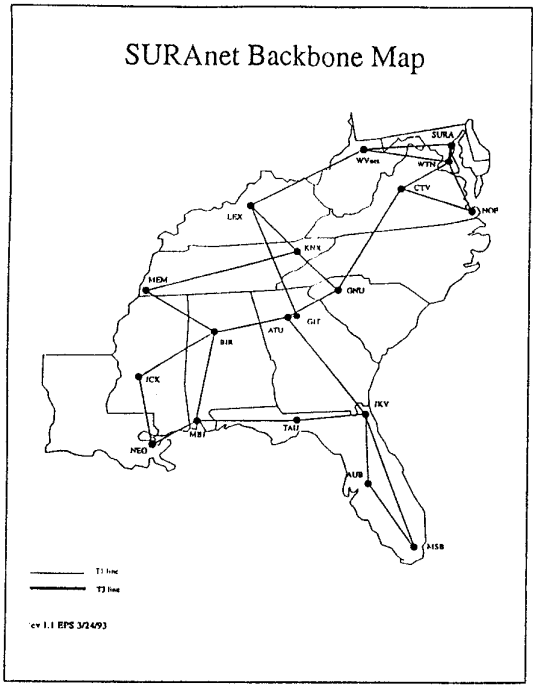

Figure 1. SURANET topology

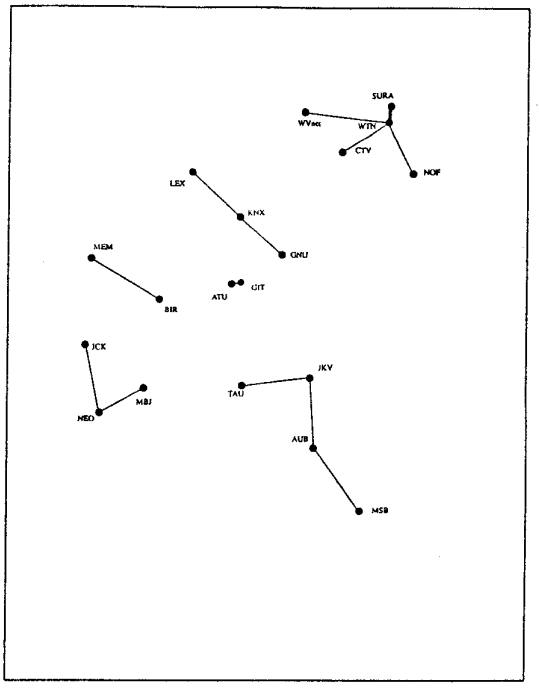

Figure 2. State after first iteration of algorithm

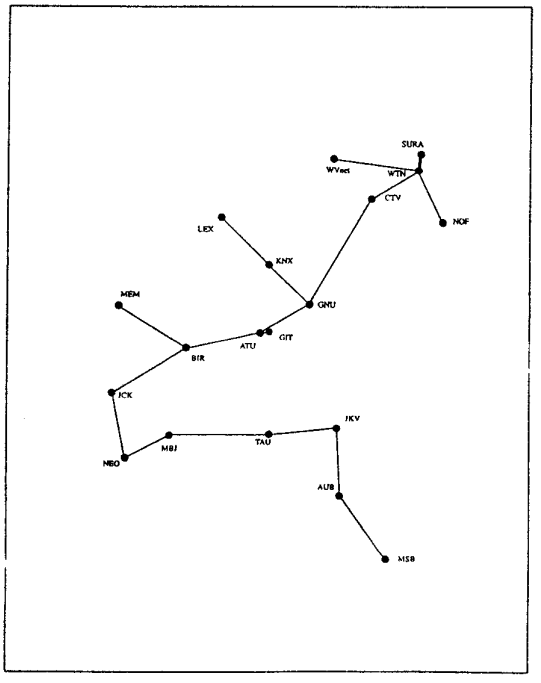

Figure 3. MST built after second iteration

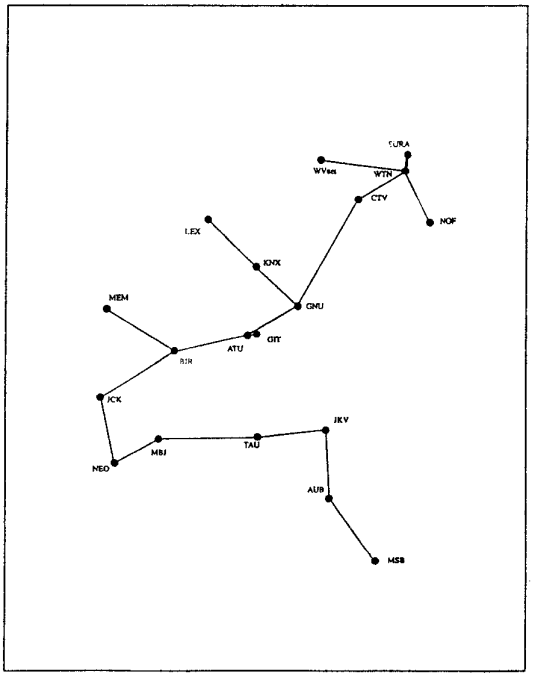

Figure 4. State while node *MSB* is in the group

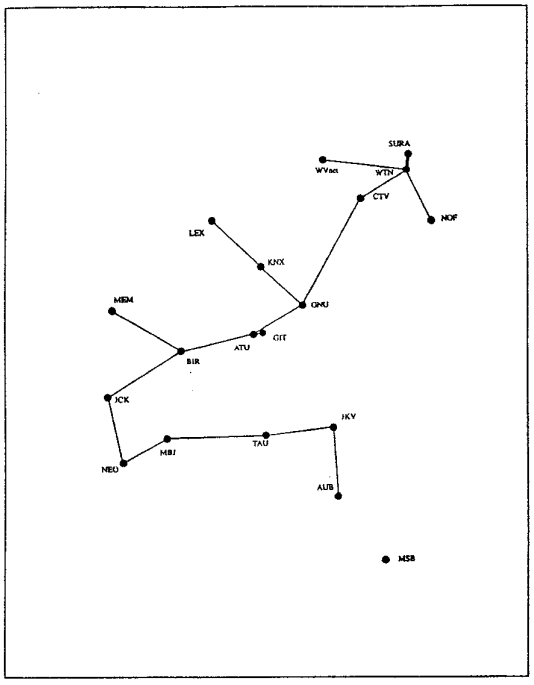

Figure 5. State after node *MSB* has exited the group

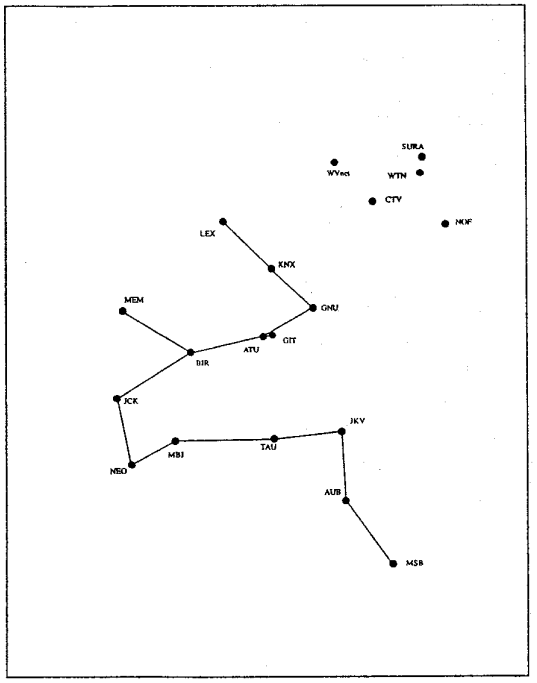

Figure 6. State before nodes join the group

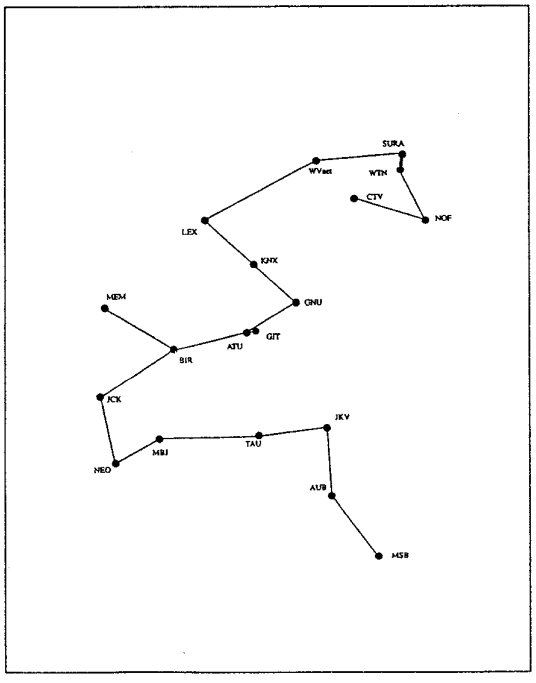

Figure 7. State after nodes have joined the group

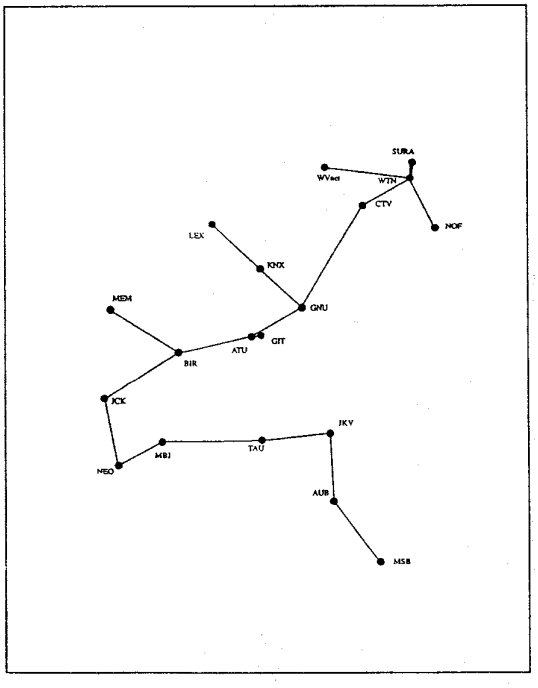

Figure 8. State after algorithm in subsection 4.1 is run (MST)