Pergamon

PII: S0305-0548(96)00074-3

Computers Ops Res. Vol. 24, No. 6, pp. 521–527, 1997
© 1997 Elsevier Science Ltd
All rights reserved. Printed in Great Britain
0305-0548/97 $17.00+0.00

DISTRIBUTED MULTICAST ROUTING IN POINT-TO-POINT NETWORKS

Jože Rugelj†‡[1] and Sandi Klavžar§[2]

[1] Department of Digital Communications and Computer Networks, J. Stefan Institute, Ljubljana, Slovenia
[2] Department of Mathematics, PEF, University of Maribor, Slovenia

(Received August 1995; in revised form August 1996)

Scope and Purpose—Multicast routing refers to delivery of the same message from a source node to an arbitrary number of destination nodes. Multicasting in point-to-point networks can be implemented as a virtual multipoint connection consisting of a set of point-to-point connections with an appropriate routing mechanism implemented in the nodes.

The problem of determining a minimum cost multipoint connection which can be modelled as a minimum cost connected subgraph that spans a given subset of nodes, is known as the Steiner tree problem in graph theory. As there is no central node with global information about the network connections in wide-area point-to-point networks, execution of the algorithm for a Steiner tree construction has to be distributed across the network nodes in such a way that each node can contribute its local information about the network connections.

This article describes a distributed heuristic algorithm for the construction of a minimum cost multipoint connection in point-to-point networks that substantially surpasses earlier used algorithms as regards time complexity.

Abstract—A new mechanism for effectively routing packets from a source to multiple destinations in large point-to-point communication networks is presented in this article. As there is no central node with the complete knowledge of network topology, and therefore conventional Steiner tree algorithms can not be used, the need for a distributed approach emerges where each node is supposed to know only the topology of its vicinity.

A distributed algorithm based on the cheapest insertion heuristics is proposed in this article. Simulation results have shown that the efficiency of the proposed distributed algorithm is practically identical to that of the distributed Kou-Markowsky-Berman algorithm, whereas it substantially surpasses DKMB as regards time complexity. © 1997 Elsevier Science Ltd

1. INTRODUCTION

Multicasting is an important mechanism for group communications in communication networks, i.e. in cases when the same information must be sent from a single source to more than one destination. It reduces transmission overheads for the sender since only one copy of each packet carrying the information needs to be sent. At the same time, it reduces the overall network traffic as only one copy of each packet passes particular network connections.

For networks in which all hosts share a common transmission channel, such as bus, ring, or satellite networks, the multicast capability can be provided trivially and, at least from the network point of view, at the same cost as unicasting. We considered the more complex case where local networks, or individual hosts, are interconnected by a point-to-point network consisting of a number of packet switches and connections between them.

The multicast capability in point-to-point networks can be implemented as a virtual multicast connection consisting of a set of point-to-point virtual connections, above one or more physical layer connections, between nodes from the multicast group, and with an appropriate routing mechanism implemented in the nodes.

There are two main optimality criteria to evaluate the goodness of a route from a given source to a set of destinations. The first relates to minimization of destination cost, which is a measure of the average delay experienced by each destination. We were more interested in the second, the network cost. In this case, the utilization of network resources is considered. The problem of determining a minimum cost

† To whom all correspondence should be addressed (e-mail:joze.rugelj@ijs.si).

‡ Jože Rugelj received his B.S., M.S., and Ph.D. degrees from the University of Ljubljana. He is a researcher at the J. Stefan Institute in Ljubljana, Slovenia, in the Department of Digital Communications and Computer Networks and Assistant Professor at the University of Ljubljana. From 1989 to 1990 he was a visiting scientist at the Joint Research Centre of the Commission of the European Communities in Ispra, Italy. His research interests include communication protocols, group communications and computer supported cooperative work.

§ Sandi Klavžar received the B.S., M.S., and Ph.D. degrees from the University of Ljubljana. He is currently Associate Professor at the University of Maribor. His research interests include graph theory, algorithms, and mathematical chemistry.

connected network that spans a given subset of nodes is known as the Steiner tree problem. For a review of this problem and the algorithms for solving it we refer to Winter [1] and Ravi [2].

2. STEINER TREE ALGORITHMS AND MULTICAST CONNECTIONS

It is well known that the Steiner tree problem is NP-complete [3]. There are a number of heuristic algorithms for deriving solutions to the Steiner tree problem in graphs that run in polynomial time, cf. [1]. The Kou-Markowsky-Berman (KMB) algorithm [4] and the Rayward-Smith (RS) algorithm [5] are typical heuristics. They are based on two different concepts. While the KMB algorithm is based on the stepwise growth of a single tree, the RS algorithm deals with a forest of smaller trees, which are gradually joined into bigger trees. Both standard algorithms have been modified by Jiang [6] in such a way that link capacities were integrated into the conventional algorithm.

The approximation algorithms described above are not practical for very large networks, since they assume a complete knowledge of the entire network in the single node where they are executed, and do not operate in a distributed fashion. When the algorithm is executed in a centralized fashion, we can assume that all the information needed is available at that place. In the distributed version, a set of node algorithms is executed in the selected network nodes. All nodes that will be included in a tree should be notified about it in advance, provided with the required initial parameters, and the execution of node algorithms should be started. This is practically not possible for routing trees based on Steiner trees since, except for the terminal nodes from a multicast group, we do not know which subset of nodes will be involved.

Therefore, we tried to find an algorithm that could take into account the limited availability of information about the global network in particular network nodes. As it turned out, cf. Jaffe [7], the minimal information that is needed by nodes to do any sensible routing calculation consists of the distance to each destination, as well as the first node on the shortest path toward the destination. This information is called *local information.*

RS type algorithms are based on merging of smaller subtrees into a larger one in each step. This method allows a high degree of parallelization, but it is impractical for distributed implementation. The problem of notification and activation of a limited subset of nodes for distributed algorithm execution is better solved in the second group of Steiner tree algorithms, which are based on the growth of a single tree. After careful examination of known algorithms for Steiner tree construction based on single tree growth, we found that the cheapest insertion (CI) heuristics, introduced by Takahashi and Matsuyama [8], best suit distributed environments in point-to-point networks. We will describe its distributed implementation in the next section and compare it in the last section with the distributed implementation of the KMB algorithm, as a typical representative of this type of algorithm.

3. DISTRIBUTED CHEAPEST INSERTION ALGORITHM (DCI)

The CI algorithm for constructing Steiner trees is a straightforward extension of the classical Prim algorithm for obtaining minimum spanning trees [9]. The algorithm begins with one singleton tree and iteratively connects the nearest unconnected terminal node to it, until all terminal nodes are connected.

In point-to-point networks, where in general particular pairs of nodes cannot communicate directly, it is inevitable that the distributed tree construction should be based on a single tree growth. The existing partially built tree is used for the distributed decision making and for the dissemination of information about the construction process. The node, selected in each phase, is activated only after it has been connected to that tree.

We next formalize the DCI algorithm by means of which a multicast virtual connection is set up. A communication network is modeled by a *network* (V, E, c), where (V, E) is a simple, undirected, connected graph, with a set of nodes V and a set of connections E and $c{:}E \rightarrow \mathrm{IR}^+$ is a cost function. Let $d(v)$ denote the degree of a node v and let $\Delta(G)$ be the largest degree of a graph G. We shall denote the depth of a tree S by *depth* (S). The *distance* between nodes $u, v \in V$, $d(u, v)$, is equal to the minimum length of a path between u and v, where the length of a path P is the sum of costs of the connections of P. By T, $T \subseteq V$, we will denote the set of *terminal* nodes, which represent entry points where members of a multicast group are connected to the network. A *routing tree* is a rooted subtree of the network (V, E, c). Its root is any of the terminal nodes; it contains all the other terminal nodes, and a set of intermediate nodes I, $I \subseteq V - T$. The nodes in the subset I belong to paths between terminal nodes. Nodes in I with $d(v) > 2$ are called *Steiner nodes* and play an active role in multicasting. In the algorithm, let U be the set of terminal nodes which are not yet selected, and let S stand for the set of connected terminal nodes and adopted intermediate nodes on the paths between already connected terminal nodes. For $v \in S$ let $sons(v)$ be the set of sons of v in the growing tree. Let $s \in T$ be a root node.

The global DCI algorithm is implemented as a set of node algorithms which are executed in the activated nodes. Here we describe the distributed algorithm as a parallel program which represents its global behavior in the network.

4. DISTRIBUTED CHEAPEST INSERTION ALGORITHM

```
Input: source node $s$, set of terminal nodes $T$,
       local information on neighboring connections in each particular node
Output: virtual multicast connection
Procedure send_p(destination, distance, nearest, proposer); and
Procedure receive_p(destination, distance, nearest, proposer);
  (*gathering the information about the not yet selected node that is nearest to the subtree*)
Procedure send_s(destination, nearest, proposer);
Procedure receive_s(destination, nearest, proposer);
  (*the dissemination of selected node across partially built tree*)
Procedure send_c(destination, nearest, unselected);
  (*activities required to activate selected nearest node and intermediate nodes on the path to it*)
begin
  $U \leftarrow T - \{s\}$; $S \leftarrow \{s\}$;
  while $U \neq \emptyset$ do
    (*each node locates the nearest unselected node, $f(v)$, and the distance to it, $l(v)$; $p(v)$ contains
    identification of the proposer*)
    foreach $v \in S$ do in parallel
      $l(v) \leftarrow \min\{d(v,w); w \in U\}$;
      $f(v) \leftarrow w$, where $w \in U$ fulfills $d(v,w) = l(v)$;
      $p(v) \rightarrow v$;
    end foreach;
    (*selection of a node is made as the info wave propagates from leaves toward the root of
    partially built tree *)
    foreach $v \in S$ do in parallel
      foreach $sons(v)$ do in parallel
        (*each node collects selections made in the subtrees, rooted in sons*)
        receive_p($sons(v)$, $l(sons(v))$, $f(sons(v))$, $p(sons(v))$);
        if $l(sons(v)) < l(v)$ then
          $l(v) \leftarrow l(sons(v))$;
          $f(v) \leftarrow f(sons(v))$;
          $p(v) \leftarrow p(sons(v))$;
        endif
      end foreach;
      (*selection in the subtree is communicated to father, except from the root*)
      if $v \neq s$ then send_p($father(v)$, $l(v)$, $f(v)$, $p(v)$);
    end foreach
    $U \leftarrow U - f(s)$;
    (*bounced info wave, originating in the root, announces which node was selected in this step*)
    foreach $v \in S$ do in parallel
      if $v \neq s$ then receive_s($father(v)$, $f(s)$, $p(s)$);
      foreach $sons(v)$ do in parallel send_s($sons(v)$, $f(s)$, $p(s)$);
      if $p(s) = v$ then
        $u \leftarrow v$; $w \leftarrow f(s)$;
        (*all intermediate nodes on the path to the selected node are activated*)
        while $u \neq w$ do
          $sons(u) \leftarrow sons(u) \cup \{first(u, w)\}$;
          send_c($first(u, w)$, $w$, $U$);
          $S \leftarrow S \cup \{first(u, w)\}$;
          $u \leftarrow first(u, w)$;
```

```
39             end while
40           end if
41         end foreach
42       end while
43 end.
```

The node algorithm running in the activated nodes in each phase, according to its local information, finds *f(v)*, a non-selected terminal node which is closest to the node, and *l(v)*, the distance to that node. *p(v)* contains identification of the proposer (*l. 4–9*). Each node then participates with its candidate (*f(v)*) in the global distributed decision; it is implemented as an information wave generated in the leaf nodes of a partially built subtree and propagating towards its root (*l. 11–25*). By the term information wave we denote a sequence of messages that are being sent across the network connections of a subtree. In our representation of the algorithm they are represented by pairs of *send* and *receive* procedures. The father node collects the data about the selections from its sons (*l. 16*). Comparing these values and its own selection, it chooses a terminal node that has the cheapest connection to the subtree (*l. 17–20*) and communicates its identity and connection cost towards the root, to its father (*l. 24*). When this information wave reaches the root, the decision made by the root represents the final selection.

Since there is no father node for the root, the wave bounces back, in the direction of the leaves, and on its way informs all activated nodes about the global decision, i.e., which node will be connected in the current step (*l. 28–30*). When the node whose candidate has been selected learns the decision (*l. 31*), it must do all that is necessary to connect and activate its candidate as well as all intermediate nodes on the path to the candidate (*l. 32–39*).

The application of DCI to a seven node network with four terminal nodes is illustrated in Fig. 1. As exactly one node is connected to the growing tree at each step of iteration of the DCI algorithm, the tree is constructed in three iterations. The behavior of the algorithm is best characterized by the sequence of messages exchanged during the construction (Fig. 2). *send_p* messages represent an information wave propagating towards the root in a distributed selection phase and *send_s* messages represent the communication of the selected node. Finally, *send_c* messages are used for the activation of the selected nodes and the nodes on the path to it from the partially built tree.

After the virtual connection (or at least a part of it) is established, the multicast routing mechanism is simple. Each node forwards each incoming packet with the multicast connection identifier to all point-to-point connections that are elements of the multicast connection, except to the incoming one.

In the next theorem we summarize the important properties of the DCI algorithm. We will assume that *send_p*, *send_s*, and *send_c* are performed in unit cost time.

Theorem 1

(i) Procedure DCI is a distributed version of the CI algorithm and uses only local information.

(ii) The computed tree S is no more than $2(1-1/|T|)$ times more expensive than an optimal Steiner one.

(iii) The time complexity of DCI is bounded by $O(|T|\cdot(\Delta(S)-1)\cdot\mathrm{depth}(S))$.

Proof

(i) Let *v* be an activated node and let *w* be a destination. Note first that *v* need not know the set *S*, which is used only to clarify the presentation of the procedure. Clearly, *v* uses its local information, the distances to other nodes and the first node on a shortest path to *w*, *first(v, w)*. Hence besides its local information, *v* uses only the set *U*, i.e. the subset of non-selected terminal nodes *T*. As this information

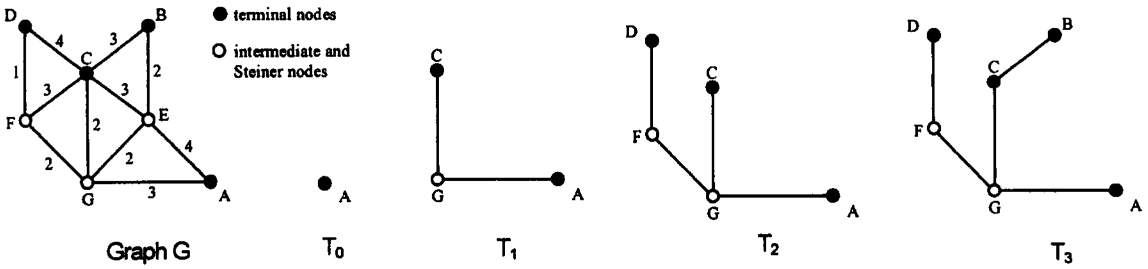

Fig. 1. The application of DCI to a seven-node graph.

T_0	A: send_c (G, C, {B,D})
	G: send_c (C, C, {B,D})
T_1	C: send_p (G, 3, B, C)
	G: send_p (A, 3, D, G)
	A: send_s (G, D, G)
	G: send_c (F, D, {B})
	F: send_c (D, D, {B}) || G: send_s (G, D, G)
T_2	C: send_p (G, 3, B, C) || D: send_p (F, 7, B, D)
	F: send_p (G, 6, B, F)
	G: send_p (A, 3, B, C)
	A: send_s (G, B, C)
	G: send_s (F, B, C)
	F: send_s (D, B, C) || G: send_s (C, B, C)
	C: send_c (B, B, {})

Fig. 2. A sequence of messages exchanged during the construction of a tree in Fig. 1.

is distributed locally along S we conclude that DCI is a distributed implementation of the CI algorithm which uses only local information.

(ii) This is a direct consequence of Theorem 1 from [8].

(iii) It is clear that the most time consuming part of the DCI procedure is the distributed selection of a node. This selection is performed $|T|$ times. In each distributed selection an activated node v computes a minimum from its neighbours' candidates, which is done in at most $d(v) - 1$ steps. The decisions of all activated nodes reach the root in *depth(S)* steps. It follows that the time complexity of DCI is bounded by $O(|T| - (\Delta(S) - 1) \cdot depth(S))$.

The upper bound from Theorem 1 (iii) in general cannot be improved, as can be seen by considering the case when S is isomorphic to a path on n vertices. Then the time complexity from (iii) reduces to $O(n^2)$, where n denotes the number of nodes in the network.

5. PERFORMANCE MEASURES

The number of messages that are exchanged between nodes during an execution of the algorithm represents a basis for estimation of the temporal complexity of the algorithm. Since the overall complexity strongly depends on the topology of a network and on the proportions between the size of the network, the number of terminal nodes, as well as on the average number of intermediate nodes on each virtual connection, it is not possible to make a trustworthy analytical assessment for the general case.

We therefore built a simulation package that allows observation of the behavior of algorithms on a large number of randomly generated networks and a comparison between different algorithms. In the simulator, we implemented the distributed CI (DCI) and the distributed KMB (DKMB) algorithm as a set of independent processes, each representing a node in the network. We observed the exchanged messages in a large number of simulation runs (500) and on different, randomly generated network topologies of small and medium size (between 16 and 30 nodes). The domains of selected parameters for the number of terminal nodes and for the density of graphs can be seen in Figs 3 and 4, respectively.

As can be seen in the figures resulting from the simulations, the DCI algorithm ranks above DKMB, as we expected due to its essentially different design approach.

An interesting result can be seen in Fig. 4, where the overall complexity of the algorithms decreases as the density of the network increases. This can be explained by the fact that more direct connections can be established and therefore less intermediate nodes are activated.

Another interesting feature is the cost of the virtual multicast connections that have been constructed by means of different algorithms. Costs in the range 1–30 were randomly assigned to individual point-to-point connections in the network. Since we are interested in the cost relations between the results of different algorithms, the absolute values are of no practical significance. Therefore we defined the multicast cost efficiency of an algorithm as the ratio between the cost of a set of point-to-point

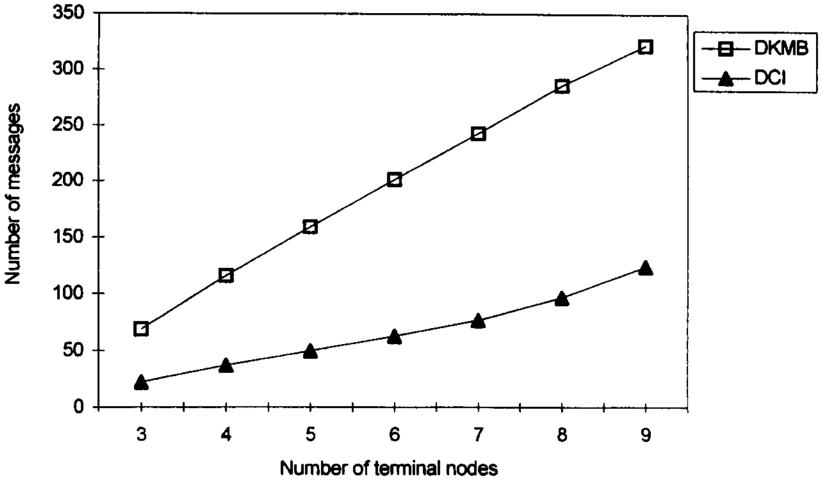

Fig. 3. Number of exchanged messages with regard to |*T*|.

connections and the cost of a multicast connection between the selected set of nodes. It can be seen that the results of both DKMB and DCI in Fig. 5 are practically identical and much better than the results of the simple distributed minimum spanning tree algorithm (DMST) [10]. In contrast, from the multicast

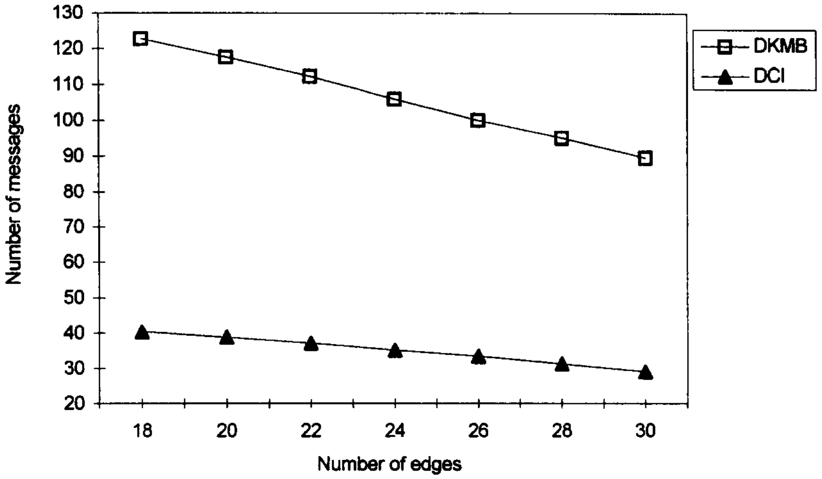

Fig. 4. Number of exchanged messages with regard to |*E*|.

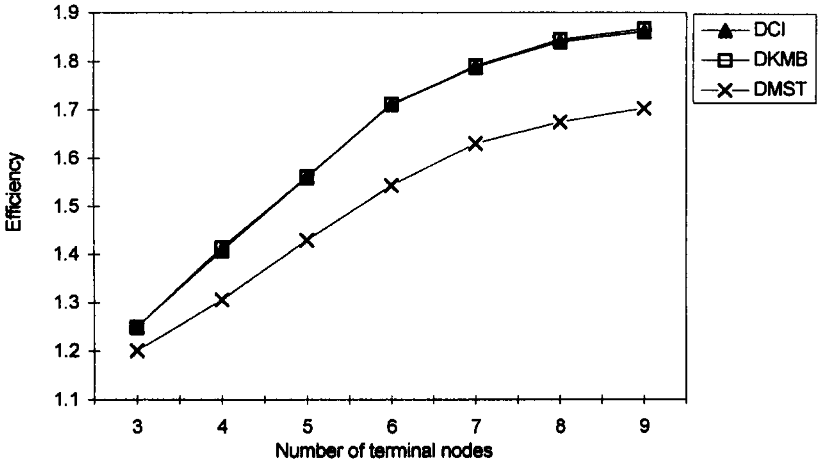

Fig. 5. Efficiency of the distributed algorithms.

connections obtained we can summarize that for the simulation parameters selected the efficiency of our algorithm is limited to 2.

As can be seen in the figures, the distributed algorithm that we developed based on the concept of the CI centralized algorithm turned out to have an attractive complexity–efficiency ratio compared with the distributed KMB algorithm.

The original algorithm can be easily upgraded to support dynamic changes in the multicast group during the multipoint connection's lifetime.

REFERENCES

1. Winter, P., Steiner problem in networks: a survey. *Networks*, 1987, **17**, 129–167.
2. Ravi, R., A primal-dual approximation algorithm for the Steiner forest problem. *Inf. Proc. Lett.*, 1994, **50**, 185–190.
3. Karp, R. M., Reducibility among combinatorial problems. In *Complexity of Computer Computations*. eds Miller, R. E. and Thacher, J.W., Plenum, New York, 1972, pp. 85–104.
4. Kou, L., Markowsky, G. and Berman, L., A fast algorithm for Steiner trees. *Acta Inform.*, 1981, **15**, 141–145.
5. Rayward-Smith, V. J., The computation of nearly minimal Steiner trees in graphs. *Int. J. Math. Ed. Sci. Tech.*, 1983, **14**, 15–23.
6. Jiang, X., Path finding algorithm for broadband multicast. In *Proc. 3rd IFIP Conf. on High Speed Networking*, 1991, pp. 201–211.
7. Jaffe, J. M., Distributed multidestination routing: the constraints of local information. *SIAM J. Comput.*, 1985, **14**, 875–888.
8. Takahashi, H. and Matsuyama, A., An approximate solution for the Steiner problem in graphs. *Math. Japon.*, 1980, **24**, 573–577.
9. Prim, R. C., Shortest connection networks and some generalizations. *Bell System Tech. J.*, 1957, **36**, 1389–1401.
10. Rugelj, J., Multicast real-time communications. In *Proc. 3rd IEE Int. Conf. on Software Eng. for Real Time Systems*, Cirencester, U.K., 1991, pp. 271–277.