© 2004 IEEE. Reprinted with permission from Proceedings of the 1st IEEE International Conference on Mobile Ad-hoc and Sensor Systems (MASS 2004), pages 519-521, October 2004.

Efficient multicast tree algorithm for ad hoc networks

Aleksi Penttinen
Networking Laboratory, Helsinki University of Technology
P.O.BOX 3000, FIN-02015 HUT, Finland
Aleksi.Penttinen@hut.fi

***Abstract*— We propose an efficient heuristic algorithm for multicast tree construction in ad hoc networks. The objective is to connect a source node to a set of destination nodes with a sequence of transmissions so that the number of transmissions is minimized. The algorithm utilizes shortest path information, which can be obtained from unicast routing tables for fully distributed implementation. In addition to multicast routing, the algorithm is applicable to stateless multicast forwarding for small multicast groups.**

I. INTRODUCTION

The source based multicast tree problem arises in the context of delivering large files or multimedia streams in a network from a source to several destinations simultaneously. Using a suitable routing tree, the resulting data transfer has a substantially lower bandwidth cost compared to the two alternatives, sequential unicast transmissions or flooding.

Multicasting in ad hoc networks differs from the traditional case of fixed networks because the nodes may use omnidirectional transmissions; all the neighbors of a transmitting node receive the same information in a single transmission. Therefore, the objective of the multicast tree problem (MTP) in ad hoc networks is defined here as to minimize the number of transmissions in the multicast tree.

Two important special cases of MTP, namely the problems with one destination (unicast) and all destinations (broadcast), have previously been treated separately in the literature. In graph theory these are known as the shortest path (SP) problem and the minimum connected dominating set (MCDS) problem, respectively. While the shortest path between any two nodes can be found in polynomial time by well-known algorithms such as Dijkstra's [2], finding the minimum connected dominating set remains NP-hard even in unit disk graphs [3].

Research on both broadcasting protocols and MCDS heuristics in ad hoc networks has been very active, because broadcasting is applied widely in the networks for information discovery and update. In [4] the authors present an extensive overview and comparison study on the broadcasting methods proposed for ad hoc networks.

Somewhat surprisingly, the general MTP has received much less attention than its special case. Instead of efficiency the focus of the multicast research has been on maintaining large dynamic multicast groups in mobile environments for which many protocols have been proposed, cf. [5]. On the other hand, multicast trees have been brought forward to minimize energy consumption in energy-constrained networks [6], [7].

In this paper we address the multicast tree problem and present a highly efficient multicast tree construction method for ad hoc networks. The algorithm is amenable for distributed implementation as each node participating in the multicast tree can decide its next hop downstream nodes in its branch independently of the other tree nodes based on unicast routing information. The crux of the method is in this distributed process of selecting the next hop forwarding node(s) so that the overall number of transmissions is minimized in the multicast tree. The algorithm gives a shortest hop path in the unicast case, is highly efficient for multiple receivers, and outperforms even specialized algorithms in the broadcast case.

II. MULTICAST TREE ALGORITHM

In this section we describe in detail the distributed method to construct efficient trees from a source to any number of receivers, including the unicast and broadcast cases. Centralized operation of the algorithm is similar to the distributed, except that all the required information can be found from the all-pairs shortest path matrix.

A. Forwarding nodes and splitting

The multicast tree is generated by a series of consecutive but independent routing decisions. The algorithm starts from the multicast source and proceeds as follows. When a node needs to find a route to a set of destinations, it divides the routing problem, the destination set, into independent subproblems each of which is resolved by a neighboring node, as if it itself was the source of the subset problem. In other words, each node decides independently the next hop neighbor(s) in the tree, i.e. the forwarding nodes and their destination sets, based on its own destination set. We call this operation as split. Figure 1 illustrates the idea. Our key contribution in this paper is the following heuristic split algorithm which has been found to perform well in practice.

B. A split algorithm

Assume that each node participating in the multicast, as a source or relay, knows the (exact or estimated) shortest hop distances between itself and the destinations, and between its neighbors and the destinations.

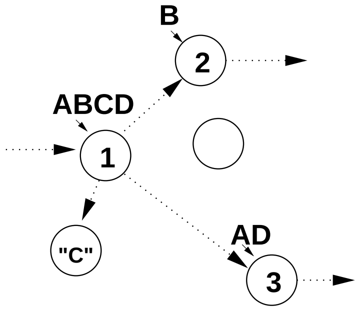

Fig. 1. Node 1 has to find a multicast tree to the destinations A,B,C and D. A split algorithm at node 1 assigns node 2 to find a multicast tree (path) to B and node 3 to find a multicast tree to A and D. C was found directly from the neighbor list.

This information might be obtained along with the unicast routing updates or by exchanging unicast information among neighboring nodes, which naturally implies some additional overhead. The required hop distances can be exchanged either pro-actively maintaining the all-pairs shortest path matrix at each node, or in on-demand fashion for the tree nodes and selected set of destinations. However, no flooding is required.

Consider the split operation $\langle i, T\rangle$ in which node i is given the task of forwarding data/finding a route to the set of destinations (aka multicast receivers) T. Let N_i be the set of neighbors of node i.

The split algorithm at i consists of finding a set of next hop forwarding nodes and divide the multicast receivers among them. During the algorithm the destinations receive assignments, either permanent or non-permanent, to neighboring nodes of i (which thus become forwarding nodes). Once made, a permanent assignment remains fixed, but any non-permanent assignment can be replaced (only) by a permanent one.

Since a transmission of i reaches all the neighboring receivers they are first removed from T. Then the algorithm repeats the following split step until all receivers in T have non-permanent assignments or T becomes empty: Let $B_j(T) \subseteq T$ denote the set of remaining receivers, which can be reached from node $j \in N_i$ with one less hop than from i. In other words $B_j(T)$ is the set of receivers to which a shortest path from i goes via j. Select a forwarding node $m = \arg\max_{j\in N_i} |B_j(T)|$, i.e. the neighbor j for which the number of elements in the set $B_j(T)$ is the largest.

The assignment works as follows. The nodes (destinations) in $B_m(T)$ are assigned permanently to m and removed from T. After that any receiver j remaining in T for which the distance from m to j is equal to the distance between i to j are assigned non-permanently to m without removal from T.

Now only the receivers for which the hop distance would have been increased if routed through m remain in T without an assignment. In case such nodes exist, we repeat the split step until all receivers in T have non-permanent assignments to a forwarding node or T becomes empty. As a result we have generated a set of split problems with forwarding nodes and their assigned destinations.

Note that the existing non-permanent assignments at the end of the split algorithm mean that the distances to the corresponding receivers are not decreased in this split. This is intentional, otherwise the resulting tree would be only a shortest path tree, yet optimized. Following the procedure, the non-permanent assignments remain with the nodes which are responsible for a larger number of destinations, but only if no other forwarding node is on a shortest path to the corresponding multicast receiver.

Obviously, if there is only one multicast destination, the split algorithm selects the next hop node from one of the neighboring nodes on a shortest path to the destination. In practice, this part of routing could also be left to the underlying unicast protocol.

The algorithm is summarized in a pseudo-code implementation in Algorithm 1. In this implementation the non-permanent assignments overlapping with permanent ones are removed in a separate loop.

Algorithm 1 Split operation, input: $\langle i, T\rangle$; forwarding node i, set of receivers T

```
$A, M, F_0 \leftarrow \emptyset$
$T \leftarrow T \backslash N_i$
/* $D_i(j)$ denotes the distance between $i$ and $j$ */
$B_j^0(T) \leftarrow \{t \in T \mid (D_i(t) - D_j(t)) = 0\},\ j \in N_i$
$B_j^1(T) \leftarrow \{t \in T \mid (D_i(t) - D_j(t)) = 1\},\ j \in N_i$
while $T \backslash F_0 \neq \emptyset$ do
    $m \leftarrow \arg\max_{j\in N_i} |B_j^1(T)|$
    /* add to beginning */
    $M \leftarrow \langle m, B_m^1(T) \cup (B_m^0(T) \backslash F_0)\rangle \cup M$
    $F_0 \leftarrow F_0 \cup B_m^0(T)$
    $T \leftarrow T \backslash B_m^1(T)$
end while
/* remove redundant non-permanent assignments */
for each $\langle m, T_m\rangle \in M$ do
    $T_m \leftarrow T_m \backslash A$
    $A \leftarrow A \cup T_m$
end for
Output $M$ /* <node, destination list> - pairs */
```

III. SIMULATIONS

We study the performance of the proposed algorithm in a series of network scenarios. For each scenario, defined by the triple (n, R_{max}, m), where n is number of nodes, R_{max} is the transmission range and m is the number of multicast receivers, the results are averaged over 500 instances of random networks in unit square. In the simulations we measure the total number of transmissions and compare the results to the optimum solutions obtained by the exact solution obtained by exhaustive enumeration.

Figure 2 shows a comparison between the proposed algorithm and the a concatenation of shortest paths from the source to the receivers (shortest path tree), when $R_{max} = 0.286$. Parameter values $m = 1$ and $m = 49$ correspond to the unicast and broadcast cases, respectively.

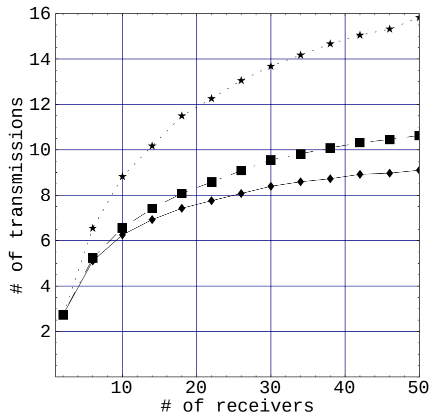

Fig. 2. Multicast efficiency with $R_{max} = 0.286$ and $n = 50$. Shortest path tree (star), proposed algorithm (square) and optimal tree (diamond).

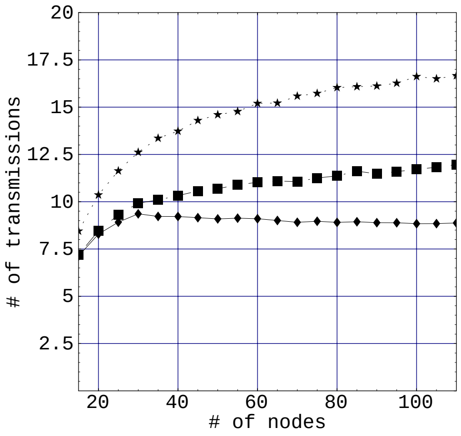

Fig. 3. Broadcast efficiency and redundancy with $R_{max} = 0.286$. CDOM [8] (star), proposed algorithm (square), and optimal MCDS (diamond)

In the broadcast case, when all the nodes belong to the multicast group, we compare the performance of the algorithm to the optimum and to CDOM, which is a (centralized) heuristic method to approximate MCDS with a constant performance ratio bound of 10 in unit disk graphs [8]. In [3] the authors claim that it can be actually shown to be 8, which is the best known to our knowledge. The required maximal independent sets in CDOM are selected in a greedy fashion.

The broadcast performance of the algorithm is visualized in Figure 3, where the number of nodes is varied between $n = 15$ (avg. node degree 3.3) and $n = 110$ (avg. node degree 21.4).

The results show the high performance of the algorithm in the multicast cases. Even in the broadcast case as it approximates MCDS significantly better than the specialized algorithm CDOM, which exhibits almost exactly the same performance as a closely related heuristics in [9] (not shown in the figure). All of the discussed methods clearly outperform flooding which would require n transmissions.

Note that in the broadcast scenarios the network parameters conform to those used in the comparison study [4] (and Figure 3 therein). There the authors reported that the best ad hoc broadcast method in their tests, AHBP, used 18% of the nodes to rebroadcast in the 110 node network. As seen in Figure 3, our algorithm uses less than 12 transmissions for the tree which means that it requires 10% of the nodes to rebroadcast the original transmission, whereas the optimum is 7%.

IV. CONCLUSIONS

We have presented and analyzed a multicast tree algorithm that utilizes shortest path information to generate highly efficient multicast and broadcast trees in ad hoc networks.

The algorithm can be applied to multicast routing and, for small receiver groups, to stateless multicast forwarding due to its fully distributed nature. In the applications the shortest path information is obtained from the unicast routing tables. Accordingly, the main disadvantage of the method may be the increased signaling load as the nodes have to exchange (at least partial) routing distance tables with their neighbors if the distances cannot be obtained directly from unicast routing updates. Furthermore, the efficiency of the algorithm depends on the validity of the route length information, which sets some constraints to the applicability of the algorithm in mobile and dynamic networks.

As the goal of the present work was to describe the tree construction algorithm and evaluate its performance, many implementation related questions remain still open for further study. For example, co-operation with unicast routing protocols in information updates and the effect of incomplete or inaccurate unicast information on multicast performance require some further attention.

ACKNOWLEDGMENTS

This work was funded by Finnish Defence Forces Technical Research Center and Graduate School in Electronics, Telecommunications and Automation (GETA).

REFERENCES

[1] S. Ramanathan, “Multicast tree generation in networks with asymmetric links,” *IEEE/ACM Transactions on Networking*, vol. 4, no. 4, Aug. 1996.

[2] D. Bertsekas and R. Gallager, *Data Networks*, 2nd ed. Prentice Hall, 1992.

[3] K. M. Alzoubi, P.-J. Wan, and O. Frieder, “Distributed heuristics for connected dominating sets in wireless ad hoc networks,” *Journal of Communications and Networks*, vol. 4, no. 1, Mar. 2002.

[4] B. Williams and T. Camp, “Comparison of broadcasting techniques for mobile ad hoc networks,” in *Proc. of MobiHoc’02*, June 2002, pp. 194–205.

[5] Z. J. Haas, J. Deng, B. Liang, P. Papadimitatos, and S. Sajama, “Wireless ad hoc networks,” in *Encyclopedia of Telecommunications*, J. Proakis, Ed. John Wiley, Dec. 2002.

[6] J. Wieselthier, G. Nguyen, and A. Ephremides, “Energy-efficient broadcast and multicast trees in wireless networks,” *Mobile Networks and Applications*, vol. 7, no. 6, pp. 481–492, Dec. 2002.

[7] A. Penttinen and J. Virtamo, “Improving multicast tree construction in static ad hoc networks,” in *Proceedings of IEEE LCN 2003*. IEEE, oct. 2003, pp. 762–765.

[8] M. Marathe, H. Breu, H. Hunt III, S. Ravi, and D. Rosenkrantz, “Simple heuristics for unit disk graphs,” *Networks*, vol. 25, pp. 59–68, 1995.

[9] R. Gandhi, S. Parthasarathy, and A. Mishra, “Minimizing broadcast latency and redundancy in ad hoc networks,” in *Proc. of MobiHoc’03*, June 2003, pp. 222–232, corrected version: http://crab.rutgers.edu/∼rajivg/pubs.html.