Low-cost ATM Multicast Routing with Constrained Delays

A. Gill Waters and John S. Crawford

Computing Laboratory
University of Kent at Canterbury
Canterbury, Kent, CT2 7NF
England

Abstract. An increasing number of networking applications involve multiple participants and are therefore best supported by multicasting. Where a multicast application consumes high bandwidth, it is important to minimise the effect on the network by offering economical multicast routes. Many new applications made possible by networks based on ATM involve real-time components and are therefore also delay-sensitive. This paper discusses reasonably simple techniques for multicast routing which tackle both of these constraints, that is: first, the route makes efficient use of network resources and, secondly, delays to all recipients are kept within a bound. The problem is NP-complete, so we present heuristics which build up a directed graph containing potential routing solutions and use a greedy approach to select a solution from that graph. The heuristics are discussed and evaluated and are shown to offer good results for a variety of situations including both large and small multicast groups. Our approach is also compared with a previous solution to this problem, which has a greater time complexity.

1 Introduction

Of the services envisaged for B-ISDN [13], many involve point-to-mulitpoint working. This is true not just for distribution services such as video and/or high definition TV, but also for a wide variety of interactive services including multimedia conferencing and distributed systems. These services must be properly supported at all levels in the protocol hierarchy by the network, using mechanisms which are both efficient and flexible [23].

An important support mechanism at the network layer is that of multicast routing. Assuming that the nodes are able to replicate packets or ATM cells onto appropriate outgoing links, multicast routing in integrated services broadband networks of generalised topology involves setting up a suitable tree using the nodes and links of the network. A good multicast routing strategy must consider several aspects of the service requirements and network capabilities and has two principal goals: first to offer facilities appropriate to each specific application and secondly, to make effective use of networking resources.

Multipoint services vary widely, imposing differing demands on multicast routing mechanisms. Some can be supported by a single multicast tree regardless of

which participant is transmitting, easing the connection set-up procedure. Other applications, for example video distribution and multi-media conferencing will need large amounts of bandwidth and are also sensitive to delays. For multi-media conferencing, different multicast trees may be necessary for each multicast source in order to keep transmissions within acceptable delay bounds. Thus, to offer the necessary flexibility, routing algorithms must take all of these requirements into account. For the most demanding services, a compromise between efficient network use and low delay (usually an upper bound on delay) is needed. This compromise forms the basis for the heuristics discussed in the paper. Other services may be able to use simpler existing techniques; a comprehensive strategy would offer the most efficient and appropriate solutions depending on the application type.
The rest of the paper is arranged as follows. In section 2, existing techniques both for the Internet and the more general case are discussed. Section 3 presents a definition of the graph-theoretical problem which is tackled here, namely the bounded delay, minimum cost multicast tree problem. In section 4 we present two related heuristic solutions to the problem developed by the authors. We also refer to other published algorithms that address the problem, one of which we have detailed and compare with our techniques [16]. Section 5 describes the performance evaluation environment and presents results of evaluations of the heuristics which demonstrate the relative merits of the techniques. The conclusion summarises the performance of the heuristics and suggests topics for further research.

2 Approaches to Multicast Routing

Before discussing our techniques in detail, we first review existing mechanisms for multicast routing, most of which deal with specific application requirements and have a single constraint, usually either minimising cost or minimising delay. The majority of work on multicast routing has been undertaken in two different contexts. These are firstly, support for host groups on the Internet and secondly, more theoretical approaches to multicasting in networks of arbitrary topology. This section discusses both of these bodies of work.

2.1 Internet Host Groups

A host group on the Internet [6] is the collection of hosts which should receive packets sent to the group's address. The philosophy behind the use of multicasting in the Internet is an extension of the connectionless way of working on the Internet: each multicast packet is delivered to the members of the group with best efforts. The sender need not know which hosts are members of the group and indeed need not itself be a member of the group. This strategy is good for some applications, but cannot properly support all of the wide range of potential multicast applications for high-speed multiservice networks.

The initial multicast routing techniques assumed that groups are densely situated throughout the Internet and routing techniques are discussed by Deering and Cheriton in [7]. Distance vector routing (based on work by Dalal and Metcalfe in [5]) is used for multicast groups on the MBone. It builds a shortest path tree, but involves pruning and regrowing the tree to cope with potential new members. Link-state routing also provides a shortest path tree; it uses flooding to propagate membership information. Neither of these techniques scales well.

A number of recent proposals offer better scalability, the major contenders being Core Based Trees (CBT) and Protocol Independent Multicast (PIM). In both cases, receivers send specific join messages, making them more suitable for sparse groups. The protocols also address reliability, state management and evolution from existing protocols.

CBT [2] builds a single delivery tree for each host group regardless of which members may act as sources. This is done by directing information to the "core" (or one of several cores) of the group from where it is multicast to all members. CBT offers a simple and effective technique for some applications, but is not optimal where achieving low delay is important. It can also suffer from potential concentration of traffic around the core(s).

Protocol Independent Multicast (PIM) [8] defines Rendezvous Points (RPs) for the group, to which sources send messages for onward multicasting to the group. When delay is critical, it can switch from the RP tree to the shortest path from a particular source if needed.

To summarise, the Internet schemes above use one or more selections from: i) a single spanning tree which may have long delays, ii) a shortest path tree rooted at each source, iii) a common multicast tree for any source within the group, which, with appropriate choice of the centre node, has a bound of twice the maximum delay to any recipient [22]. In section 4, heuristics are introduced which offer a compromise between the first two, with a greatly improved delay bound on the third.

2.2 Theoretical Work on Multicast Routing

The general graph-theoretical problem for determining multicast routing trees is as follows. Given a connected graph $G = \langle V, E \rangle$ where V is the set of vertices and E is the set of edges, we wish to find a tree $T = \langle V_T, E_T \rangle$ where $T \subseteq G$ and T joins the vertices of a multicast group, M, where $M \subseteq V$. The tree will be selected based on some optimising criterion or criteria which depend on the cost(s) incurred in travelling along any edge in E. The graphical representation either ignores costs associated with the nodes (network switches) or includes these within the costs associated with the links and is an approach suitable for large scale networks of arbitrary topology. The majority of work has looked either at low-delay solutions or at low overall cost solutions.

Finding a multicast tree which minimises the *delay* to all of the recipients can be done quite simply using Dijkstra's algorithm. (See for example [1].) It can then be pruned of nodes and/or links which are not needed to reach the members of

a multicast group. The time complexity of Dijkstra's algorithm is $O(n^2)$, where n is the number of vertices in the network.
Several algorithms are available to minimise the total *cost* of a tree which includes all the nodes in a network, e.g. Prim or Kruksal, (also described in [1]). The time complexity is similar to Dijkstra's algorithm.
When the members of a multicast group, M, are a proper subset of the set V of the vertices of the graph G, finding a minimum cost solution is harder, because it must decide whether the inclusion of vertices not in M would actually lead to a cheaper solution. The problem is well known in graph theory as the Steiner tree problem which has been shown to be NP-complete but is tractable for small networks. See for example Dreyfus and Wagner [10]. Heuristics for the Steiner tree problem have been shown to approach the ideal solution of small networks. See for example Waxman [26] or Rayward Smith [19].
Other theoretical work has concentrated on slightly different aspects. Bharath-Kumar and Jaffe consider multicast trees which trade efficiency with low average delay to recipients [3]. Kadirire introduces the concept of "geographic spread" [15] which makes it more likely that a node wishing to join an existing multicast group will find a cheap path to the tree. Jiang developed a Steiner tree variation which takes account of link capacities for high bandwidth applications, prioritising choice on high capacity links [14]).

3 Defining the Bounded Delay, Minimum Cost Multicast Routing Problem

Our concern is to find suitable solutions for a number of high speed networking applications which work within two constraints, i.e they are economical within a delay bound. The bounded delay minimum cost multicast routing problem can be stated as follows.

- Given a connected graph $G = \langle V, E\rangle$ where V is the set of its vertices and E the set of its edges, and the two functions: cost $c(i, j)$ of using edge $(i, j) \in E$ and delay $d(i, j)$ along edge $(i, j) \in E$.
- Find the tree $T = \langle V_T, E_T\rangle$, where $T \subseteq G$, joining the vertices s and $M_{k,k=1,n} \in V$ such that $\sum_{(i,j)\in E_T} c(i, j)$ is minimised and $\forall k, k = 1, n$; $D(s, M_k) \leq \Delta$, the delay bound, where $D(s, M_k) = \sum_{(i,j)} d(i, j)$ for all (i, j) on the path from s to M_k in T.

As can be seen, the general case has two cost parameters, one of which represents the delay and the other the cost of using the link. For the purpose of the following evaluations, the delays on a link will be assumed to include a fixed nominal value due to queueing in the previous node. The cost function may be proportional to real costs, it may reflect available bandwidth on the link or the total bandwidth of the link or it may relate to the link's position in the network's topology. It may in some cases be proportional to the delay parameter for the link. By separating the two parameters out, the techniques discussed here will enable

network designers to incorporate their own cost strategies using these or other factors in calculating suitable multicast routes.
Note that, if the delay is unimportant, the problem reduces to the Steiner tree problem. The addition of the finite delay bound makes the problem harder, and it is still NP-complete, as any potential Steiner solution can be checked in polynomial time to see if it meets the delay bound.

4 Heuristics for Multicast Routing

For convenience we label the heuristics presented here the "Waters", "Crawford" and "Kompella et al" heuristics.

4.1 The Waters Heuristic

A simple way of finding a bounded delay multicast routing tree is to use Dijkstra's algorithm. This is the "shortest path" approach used in the Internet, but it is not optimised for the total cost of the tree. The new heuristic, the "Waters heuristic", provides an effective compromise by extending the procedure of Dijkstra's algorithm to find alternative paths which still lie within the delay bound, from which a low cost tree can be constructed.
The heuristic was first published in [24] along with some simple preliminary evaluations. This paper introduces important variations and comprehensive evaluations of these. In the first version to be introduced, the delay bound, Δ is taken to be the maximum delay from source, s, to any vertex in the network (the broadcast delay bound). Variations to this procedure will be discussed later. The procedure is as follows:

1. Use an extended form of Dijkstra's shortest path algorithm, to find for each $v \in V - \{s\}$ the minimum delay, dbv, from s to v. As the algorithm progresses keep a record of all the dbv found so far, and build a matrix $Delay$ such that $Delay(k, v)$ is the sum of the delays on edges in a path from s to v, whose final edge is (k, v). When the algorithm is complete, the maximum dbv found becomes the broadcast delay bound dbB.
2. Use dbB as the delay bound Δ. Set all elements in $Delay(k, v)$ that are greater than dbB to ∞. The matrix $Delay$ then represents the edges of a directed graph derived from G which contains all possible solutions to a multicast tree rooted at s which satisfy the delay constraint.
3. Now construct the multicast tree T. Start by setting $T = \langle \{s\}, \emptyset \rangle$.
4. Take $v \in V_T$, with the maximum dbv and join this to T. Where there is a choice of paths which still offer a solution within the delay bound, choose at each stage the cheapest edge leading to a connection to the tree.
5. Include in E_T all the edges on the path (s, v) not already in E_T and include in V_T all the nodes on the path (s, v) not already in V_T.
6. Repeat steps 4 and 5 until $V_T = V$, when the broadcast tree will have been built.
7. Prune any unnecessary branches of the tree beyond the multicast recipients.

4.2 A Worked Example

To illustrate the working of the heuristic, we take the graph shown in Fig. 1a) (also used as an example in [16]). The bracketed parameters for each link indicate $(cost, delay)$. The example finds the multicast route from source F to destinations B, D, E and H.

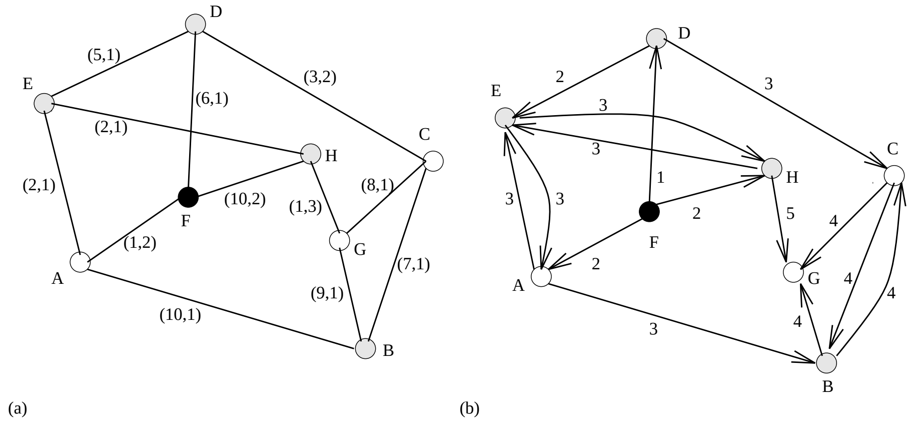

Fig. 1. Sample graph to illustrate the Waters heuristic

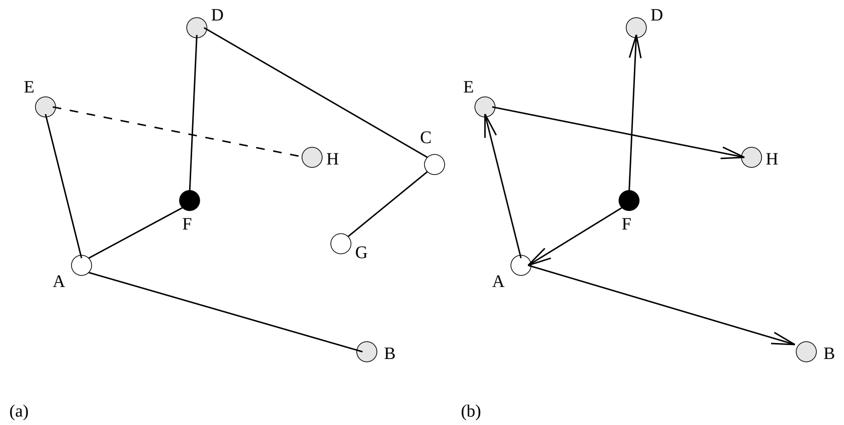

Fig. 2. Stages in construction of multicast tree

The application of the extended form of Dijkstra's algorithm results in the directed graph shown in Fig. 1b) where parameters shown against each link represent the total delay from the source, F, to reach the node at the end of that link.

The broadcast delay bound is 4 (to G). Edge HG is removed as it gives a higher delay than the bound. The tree is then constructed starting with $T = \langle F, \emptyset \rangle$. First G is connected to F using the path FD, DC, CG. B is connected via FA, AB and then E via AE, and H via EH (Fig. 2a). Finally, the edges DC and CG are pruned to give the multicast tree of Fig. 2b) whose total cost is 21 units and has a final delay bound of 4 (to H).
A minimum spanning tree, when pruned to the multicast, gives a cost of 20 units and a delay bound of 7. The tree produced by the standard use of Dijkstra's algorithm and then pruned would result in a solution with a cost of 32 units and a delay bound of 3. This simple example shows that a good compromise between delay bound and overall cost has been found by the Waters heuristic.

4.3 Time Complexity of the Heuristic

The first stage, determining the directed graph, has the same time complexity as Dijkstra's algorithm, which is at most $O(n^2)$. The vertices can be put in delay bound order during the construction of the directed graph.
In the second stage, building the multicast tree, requires a depth first search from each leaf node to find a path to the source. As the multicast tree grows the search space for each leaf to source node path becomes smaller. The time complexity of the depth first search is $O(max(N, |E|)$ [12] where N is the number of nodes, and E is the set of edges, in the leaf node to source tree. The values of N and $|E|$ depend on the topology of the network and the position of the multicast source node. Our evaluation work has shown that in practice the time required for the multicast tree construction is much closer to $O(n * m)$, where m is the maximum node degree, than $O(n^2)$ although where there are high node degree network clusters connected by very few (> 1) links, it can be much greater than $O(n^2)$.

4.4 Variations of the Waters Heuristic

A number of variations of the heuristic are possible in terms either of delay bound or choice of links to include in the multicast tree.
The tightest delay bound is the maximum of the minimum delay from source, s, to any of the destinations M_k in the multicast group. To change the heuristic to keep within this bound: first, eliminate all elements of the Delay matrix giving a greater delay and secondly, work within the multicast delay bound when constructing the tree. Without further adaptation, this method is subject to running into loops and not yielding a solution.
In setting an arbitrary delay bound, if this is between the multicast and broadcast bounds, a similar technique to the multicast bound can be used. With no bound on delay, the problem reduces to the Steiner tree problem as previously discussed. With a finite bound greater than the broadcast bound, there are two options. The broadcast bound technique described could be used, thus meeting a more stringent delay constraint than required. Alternatively, Dijkstra's algorithm could be further extended until all vertices were marked with the total

delay from the source, s, found by approaching them from every neighbouring vertex. Elements of the Delay matrix exceeding the bound would be removed. This would give further alternatives for the multicast tree and could be expected to yield a lower overall cost than using a broadcast bound.
While evaluating the Waters heuristic on realistic network topologies during his MSc project, John Crawford discovered that the broadcast bound heuristic could also produce loops while examining a multicast group using the topology of MichNet. To combat the potential problem of looping, he put forward an alternative heuristic based on the Waters heuristic but with a different choice function. This alternative is discussed in [4]. Incidentally, at the same time Salama, Reeves, Viniotis and Sheu proposed a similar variation to the Waters heuristic in their work on multicast routing algorithms [20]. The modified version (which we call the Crawford heuristic) changes the method described in section 4.1 as follows:

- At step 1, at the same time as building the $Delay$ matrix, build a corresponding matrix, $TotCost$, which gives the total cost to each vertex along the path which gives the delay in the corresponding element of the Delay matrix. i.e. If $Delay(k,j)$ represents the total delay from source s to node j, with last edge (k,j), $TotCost(k,j)$ is the total cost from the source s to node j incurred by following the same path ending in edge (k,j).
- At step 4, in constructing the path to the existing tree, instead of choosing the cheapest edge to connect to the existing tree, choose the edge which gives the cheapest total cost from the source to the vertex being considered, provided of course that the choice ensures that the delay bound is met.

An interesting and valuable property of the Crawford heuristic is that, because the total cost from the source is considered, this path will always head for the source without looping. (A loop would impose an additional and therefore greater cost.) The original Waters heuristic has now been modified to detect potential loops and remove them by using a recursive procedure whilst selecting paths back to the tree which will unwind if a loop is detected.

4.5 Heuristics with an Arbitrary Delay Bound

Several heuristics have been proposed that use arbitrary delay bounds to constrain multicast trees. The proposal of Kompella, Pasquale, and Polyzos [16], which uses a constrained application of Floyd's algorithm, is described below and used as a comparison with our heuristics. Widyono [28] proposed four heuristics based on a constrained application of the Bellman-Ford algorithm. Zhu, Parsa and Garcia-Luna-Aceves [29] based their technique on a feasible search optimisation method to find the lowest cost tree in the set of all delay bound Steiner trees for the multicast. Evaluation work carried out by Salama, Reeves, Vinitos and Sheu [21] indicate that all these heuristics have similar performance, which is generally better than the performance of our heuristics, but that the time they require to compute their solutions may be too long to be useful in large networks.

4.6 The Kompella et al Heuristic

The algorithm has three main stages.

1. A closure graph (complete graph) of the constrained cheapest paths between all pairs of members of the multicast group is found. The method to do this involves stepping through all the values of delay from 1 to Δ (assuming Δ takes an integer value) and, for each of these values, using a similar technique to Floyd's all-pairs shortest path algorithm (see [11]), which has a time complexity of $O(n^3)$. The overall time complexity of this stage is $O(\Delta n^3)$, where n is the number of vertices in the graph.
2. A constrained spanning tree in the closure graph is found using a greedy algorithm. Two alternative selection mechanisms are presented, one based solely on cost, which we used in our evaluation. The other based on cost and delay. These take $O(m^3)$ where m is the number of nodes in the multicast group.
3. The tree is regenerated in the original graph, removing any potential loops. This takes O(nm).

It should be noted that the complete graph needed for Kompella et al's solution may not exist as there will be no guarantee that the delay between any two arbitrary multicast nodes is within the delay bound. However, if these edges are set to infinity in the complete graph it will still be possible to construct a tree connecting the multicast group, provided there are paths from the source to each of the multicast nodes which fall within the delay bound.

5 Performance Modelling of the Heuristics

5.1 Network Models

Two models were used for the evaluation of the heuristics. The first is attributed to Waxman [26] and the second to Doar [9]. The Waxman model randomly distributes nodes over a rectangular coordinate grid and uses the Euclidean metric to determine the distance between them. Edges are then introduced into the network using a probability function that depends on their length.

The network model suggested by Doar is based on that of Waxman. Doar introduced a scaling factor to overcome the tendency of each node's degree to increase as the number of nodes in the network increased, a problem inherent in the Waxman model. Doar goes further by introducing hierarchical graphs as network models. These are constructed using the modified probability function to generate clusters of networks, that are then connected to a central core network using a fixed number of links. We used both network models in our evaluation work to assess the performance of our heuristics in isolated networks and networks of interconnected clusters.

We used edge lengths to represent link delay (a combination of delay attributes, e.g. propagation delay, transmission delay). Random edge costs were uniformly

generated in the range [1..50] to represent the availability of resources for the edge (e.g. bandwidth).
Evaluation of the Kompella et al solution using large networks has proved impractical with our current implementation of their algorithm, because of the time required to compute the constrained cheapest paths. For these evaluations we have used the Waxman model to generate 20 random networks of 20 nodes each. The average degree of nodes in these networks is 5. Against these we find solutions for ten multicasts of each of six multicast group sizes. For the evaluation and comparison of the Waters and Crawford heuristics we have used 200 Doar hierarchical model networks each of 100 nodes. The average degree of nodes in these networks is 4 (unless stated otherwise). For each network, ten samples of each multicast group size are used.

5.2 Performance Evaluation

It is not possible to compare our solutions with optimal solutions as finding optimal solutions to the delay bound minimal cost multicast routing problem is impractical for networks other than very small ones. We compare the cost performance of the heuristics against the costs of Prim's minimum cost spanning trees [1], after pruning them to the multicast. We compare the delay performance of the heuristics against the delay bounds of Dijkstra's shortest path trees [1], after pruning them to the multicast (i.e. the multicast bound). The delay bound used for each heuristic is of course not exceeded.

Further evaluations can be found in [25]. These are carried out using the Waxman model for networks of high node degree, but the results are comparable with the more realistic cases presented here.

5.3 Comparison of Kompella et al, Waters and Crawford

Before evaluating our heuristics under a variety of conditions, we first compare them for small networks with Kompella et al's solution.

The algorithm of Kompella et al uses an arbitrary delay bound which we set equal to the broadcast delay bound, as used by our heuristics. The cost of the Kompella et al solutions will fall as the delay bound is increased because cheaper constrained cheapest paths will become available.

Figure 3 illustrates the percentage excess cost of the multicast trees found using the three heuristics under evaluation. The multicast solutions of all three heuristics outperform Dijkstra's SP solution for cost and Prim's MST for delay.

The excess costs for Dijkstra's SP solutions rise relative to Prim's MST as the multicast group size increases, since Dijkstra will add relatively fewer common edges to reach the same nodes than does Prim as the tree grows. This characteristic is common to the three heuristics because they all use shortest paths between nodes from which to select lowest cost trees, although Kompella et al use Floyd's all pairs shortest paths algorithm [1] rather than Dijkstra's SP. Interestingly, for larger groups (see figure 3) the Waters heuristic overcomes this

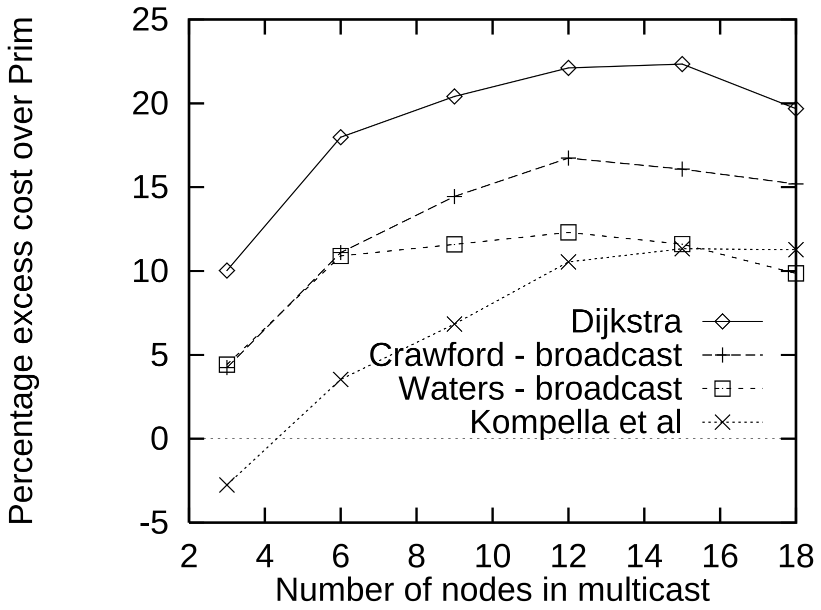

Fig. 3. Cost performance

characteristic because it has more candidate return edges to choose from when constructing return paths to the source.

The algorithm of Kompella et al generally builds multicast trees with lower costs than the solutions of Waters and Crawford, except for large multicast groups when the Waters solution becomes cheaper.

Kompella et al grows a minimum cost spanning tree from a complete graph of constrained cheapest paths between the multicast nodes by successively adding nearest cost nodes to the tree. Both the Waters and Crawford heuristics build a constrained spanning tree for the whole network, which is then pruned back to the multicast. This process in the heuristics compromises cost savings. The differences between the Waters and Crawford heuristics are examined below.

An advantage of the Kompella et al heuristic and the techniques referred to in section 4.5 is that they use an arbitrary delay bound, which can be greater than the broadcast bound or between the broadcast and multicast bounds. We are currently considering the use of arbitrary delay bounds in our heuristics (see section 4.4). The cost saving of our less time-consuming heuristics can however be seen to merit further evaluation.

5.4 Comparison of Waters and Crawford

Figure 4 illustrates the performance of the Waters and Crawford solutions for networks where edge delay and cost are independent functions.

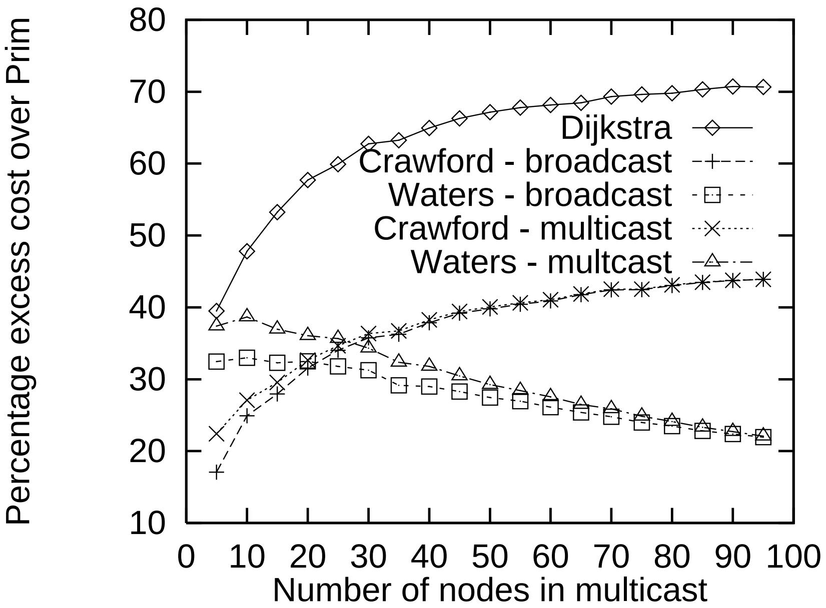

Fig. 4. Cost performance with low node degree hierarchical networks

For both the Waters and Crawford heuristics the broadcast delay bound gives cheaper solutions than does the multicast bound, because the larger bound allows a greater choice of return edges or paths than does the tighter bound.
The Crawford heuristic provides cheaper solutions for small multicast groups than does the Waters heuristic but, as the multicast group size increases relative to the network, Waters solutions become cheaper and Crawford's more expensive. The Crawford solution's excess costs increase because the heuristic mimics Dijkstra's algorithm by using costs based on the shortests paths to find return routes to the source. Its costs will generally be less than Dijkstra's because it chooses the cheapest of all the possible return paths within the delay bound, rather than the path with the least delay. The Waters heuristic constructs its low cost tree by successively adding the lowest cost edges from the highest delay nodes to build constrained paths back to the multicast source, via the existing tree, until all nodes are in the tree. As the multicast group size increases the chances of a node joining the multicast tree via its cheapest return edge will increase. As the group size increases each return path will generally have

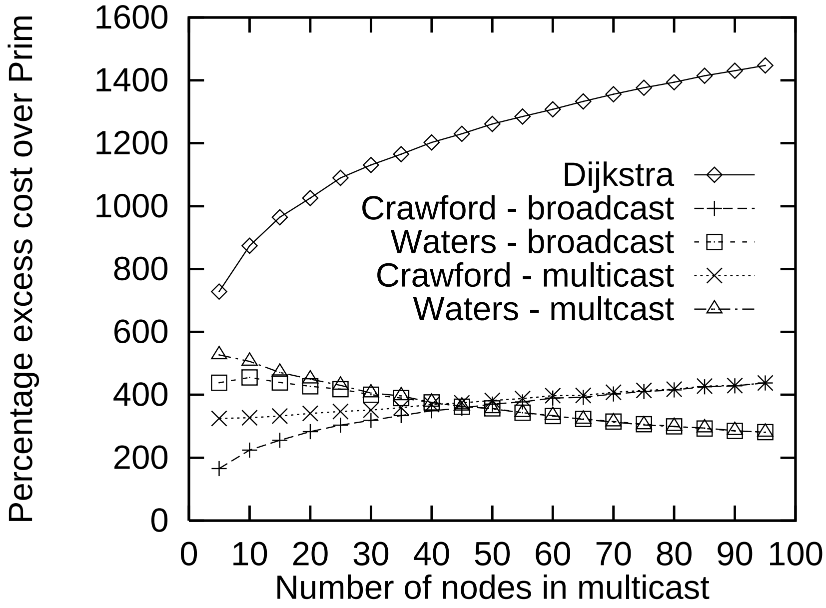

Fig. 5. Cost performance with high node degree networks

more edges, giving the algorithm more alternative paths back to the source to choose from. Both these characteristics help to significantly reduce the cost of the heuristics solutions.

The multicast group size at which the Waters heuristic begins to provide cheaper solutions than Crawford's depends on the node degree of the network. As the average node degree of the network increases the excess costs of the two algorithms begin to level out and the multicast group size at which they cross increases, as illustrated by figure 5.

When constructing the multicast trees the depth first search visited each node once only in nearly 70% of the cases for the Waters heuristic and 97.8% of the cases for the Crawford heuristic. In the worst cases the number of nodes visited by the Waters heuristic and Crawford variant were $3.44n$ and $1.71n$ respectively (where n is the number of nodes in the network). For highly connected networks (excluding dense clusters sparsely connected) the percentage of nodes visited only once during the search increased to 82.2% and 99.9% respectively.

Confidence limits at the 95% level in all graphs for mean percentage excess costs are within the range 1% to 4% for multicast group sizes in the range 20 to 95 nodes. For small multicast group sizes the variance in mean percentage excess cost increases as the proportion of mutually exclusive multicast permutations increases. Evaluations with much larger samples of smaller multicast groups give similar results within narrower confidence limits. Actual cost savings of a specific multicast depends on the topology of the network it is applied to and the position of the source node within the network.

5.5 Comparison where Edge Costs equal Edge Delays

Dijkstra gives much better cost performance when edge costs are equal to edge delays, that is where edges effectively have a single metric (figure 6). This is because, in minimising total delays to each multicast group node, it also keeps costs to each node low. Unlike Dijkstra, the Crawford solution may have a choice

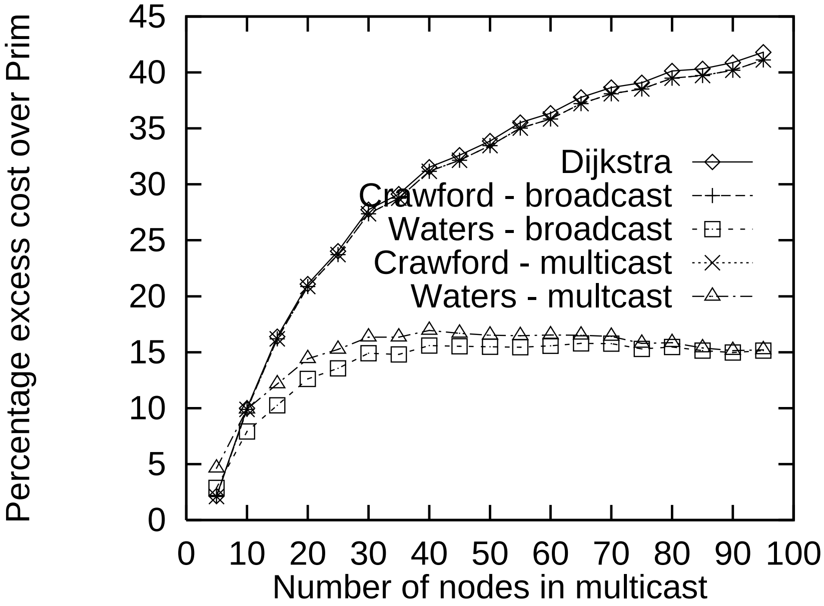

Fig. 6. Cost performance with edge costs = edges delays

of alternative return paths, albeit of the same cost, which may intersect sooner than the Dijkstra paths. It is this quality which results in Crawford's heuristic providing marginally cheaper solutions than Dijkstra's.
The Waters heuristic achieves a much lower cost than Crawford, because by choosing the cheapest constrained edges it has a greater choice of paths back to the source.

5.6 Comparison with Unit Edge Delays

In networks where delay is measured in terms of the number of "hops" between source and destination (unitary edge delay) the excess costs of all the algorithms

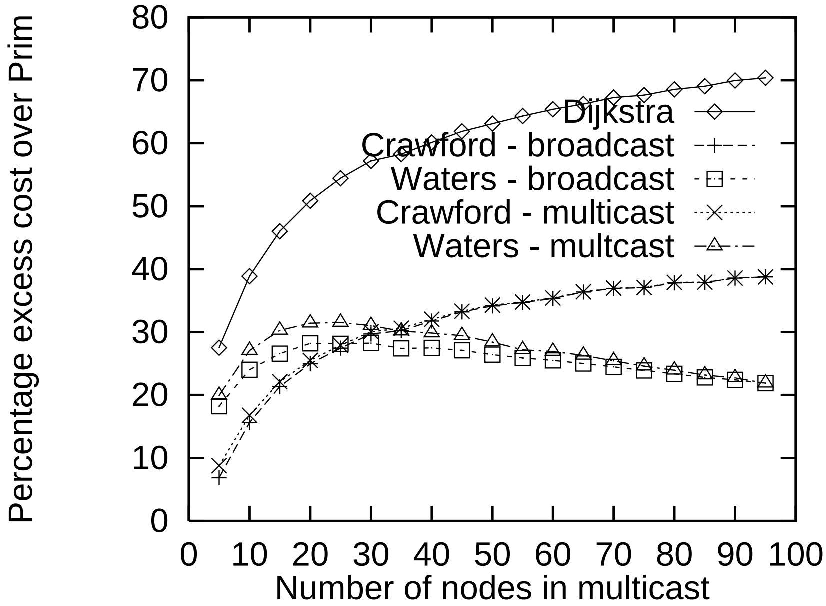

Fig. 7. Cost performance with unit edge delays

is reduced for smaller multicast group sizes (Figure 7). This is because Dijkstra's algorithm, the basis of the heuristics, will minimise the number of edges along each path when edge delay is unitary. This is not the case with non-unitary edge delays where the total path delay is minimised.

6 Scalability and Dynamicity

Scalability of multicast routing algorithms is related to both the time the algorithm requires to compute a multicast tree and the volume of state information required for the computation. Our techniques generally have low computation time, as explained in Section 4.3. The volume of state information required by a multicast routing algorithm depends on whether the multicast tree is shared or source based. Shared multicast trees are designed for use with sparse multicast groups and reduce state information requirements at the cost of increased delay. Source based multicast trees are used where delay is important or the multicast group is dense, but they use more state information. Our heuristics, being bound by an upper delay constraint are source based. A study that addresses the tradeoffs between sparse and dense mode multicasting is to be found in [27].

In practice multicast groups are often dynamic; members join and leave the multicast group thoughout its lifespan thus affecting the cost and delay of the multicast tree. A simple technique for dynamic groups is to cache the multicast tree for the broadcast group. This can be used to shrink or grow the tree easily, from any group size. We are investigating even simpler techniques such as joining a new member to the tree by the least delay path and preliminary results [17] show this to be about 75% as efficient as tree recalculation, even after 2000 modifications. tree.

7 Conclusion and Further Research

Many multipoint applications envisaged for high-speed broadband networks, which include both distribution services and interactive services, will need to make efficient use of network resources while also being sensitive to delay.

We have shown that relatively inexpensive heuristics can offer solutions which balance these needs, for a variety of conditions. The performance of these heuristics compares well with the performance of more stringent, but much more costly heuristics, when the delay bound of the multicast is at the broadcast delay bound. Variations of our heuristics constrained to the multicast delay bound show how costs rise as the delay bound becomes more restrictive.

The vast majority of cases we studied completed the calculation within $O(n^2)$ time for graphs of n nodes. However, under certain conditions, the construction of leaf to source paths can lead to an unacceptably lengthy calculation due to loops or delay bound violations. A simple check for such cases would be to set a limit on the number of times nodes are considered (say $10n$) whilst constructing the tree from the undirected graph. If this limit is exceeded, the new technique could be abandoned and the tree constructed using the pruned Dijkstra's algorithm for which all information is available. This would keep calculations within $O(n^2)$ in all cases. Work is continuing to investigate the nature of the networks and multicast groups which cause this behaviour and we hope that this will lead to more efficient solutions for these special cases. (Other published techniques also include detecting and removing loops and may run into similar problems, which we have not investigated.)

In large and widely dispersed networks the broadcast delay bound may not sufficiently constrain a multipoint application. In tightly clustered networks the broadcast delay bound may be too restrictive on a multipoint application. The multicast delay bound will always be the tightest constraint on the multicast. Variations to the way the constrained delay tree is constructed in our heuristics address this issue (see section 4.4) and are currently under study.

Our heuristics are source based and assume knowledge of the network and the multicast group. The Waters heuristic performs well for larger multicast groups, whilst the Crawford heuristic is more suited to smaller multicast groups. Integration of these heuristics into a hybrid may provide better overall solutions to the delay bound minimum cost multicast routing problem. This development is also currently under study.

We believe that the performance gains of our heuristics offer a promising technique for consideration both for wide area ATM networks (and B-ISDN) and for the Internet, which is required to accommodate an increasing number of real-time multicast applications. Consequently, we are considering how our techniques can be applied to emerging multicasting standards. Preliminary results applying our work to both Core Based Trees (CBT) and Protocol Independent Multicast (PIM) have shown that they can be used to good effect [18].

8 Acknowledgements

We would like to acknowledge the support of the UK Engineering and Physical Sciences Research Council (EPSRC) for our work on multicast routing (Grant ref. GR/K55837).

References

1. A.V. Aho, J.E. Hopcroft, and J.D. Ullman. *Data structures and algorithms*. Addison Wesley, 1987.
2. A.J. Ballardie, P.F. Francis, and J. Crowcroft. Core based trees. *Computer Communications Review*, 23(4):85–95, 1993.
3. K. Bharath-Kumar and J.M. Jaffe. Routing to multiple destinations in computer networks. *IEEE Trans. on Communications*, COM-31(3):343–351, March 1983.
4. J.S. Crawford. Multicast routing: evaluation of a new heuristic. Master's thesis, University of Kent at Canterbury, 1994.
5. Y.K. Dalal and R.M. Metcalfe. Reverse path forwarding of broadcast packets. *Communictaions of the ACM*, 21(12):1040–1048, December 1978.
6. S. Deering. Host extensions for ip multicasting. rfc 1112, aug 1988.
7. S.E. Deering and D.R. Cheriton. Multicast routing in datagram internetworks and extended lans. *ACM Transactions on Computer Systems*, 8(2):85–110, May 1990.
8. S.E. Deering, D. Estrin, D. Farinacci, V. Jacobson, C-G. Liu, and L. Wei. An architecture for wide area multicast routing. *Computer Communications Review*, 24(4):126–135, October 1994.
9. J.M.S. Doar. Multicast in the Asynchronous Transfer Mode Environment. Technical Report No. 298, University of Cambridge Computing Laboratory, April 1993.
10. S.E. Dreyfus and R.A. Wagner. The steiner problem in graphs. *Networks*, 1(3):195–207, 1971.
11. R.W. Floyd. Algorithm 97: Shortest path. *Communications of the ACM*, 5(6):345, 1962.
12. Alan Gibbons. *Algorithmic Graph Theory*. Cambridge University Press, 1989.
13. ITU-T Reccomendation I.121. B-ISDN service aspects.
14. X. Jiang. Routing broadband multicast streams. *Computer Communications*, 15(1):45–51, Jan/Feb 1992.
15. J Kadirire. Minimising packet copies in multicast routing by exploiting geographic spread. *Computer Communications Review*, 24(3):47–62, July 1994.
16. V.P. Kompella, J.C. Pasquale, and G.C. Polyzos. Multicast Routing for Multimedia Communications. *IEEE/ACM Transactions on Networking*, 1(3):286–292, 1993.
17. Kuzminski.T. Alternatives for multicast routing in atm networks. Master's thesis, University of Kent at Canterbury, 1996.
18. S. Parkinson. Multicast Routing in the Internet:Evaluating Proposed Routing Mechanisms. Master's thesis, University of Kent at Canterbury, 1995.

19. V.J. Rayward-Smith. The computation of nearly minimal steiner trees. *Int. Journal of Maths, education Science and Technology*, 14(1):15–23, 1983.
20. H.F. Salama, D.S. Reeves, I. Vinitos, and Tsang-Ling. Sheu. Comparison of Multicast Routing Alogrithms for High Speed Networks. Technical report, North Carolina State University, September 1994.
21. H.F. Salama, D.S. Reeves, I. Vinitos, and Tsang-Ling. Sheu. Evaluation of Multicast Routing Algorithms for Distributed Real-Time Applications of High-Speed Networks. In *Proceedings of the 6th IFIP Conference on High-Performance Networks (HPN'95)*, 1995.
22. D. Wall. *mechanisms for broadcast and selective broadcast.* PhD thesis, Stanford University, 1980.
23. A.G. Waters. Multicast Provision for High Speed Networks. In A. Danthine and O. Spaniol, editors, *4th IFIP Conference on High Performance Networking*, pages G1.1–G1.16. Springer Verlag, December 1992.
24. A.G. Waters. A new heuristic foir atm multicast routing. In *2nd IFIP Workshop on Performance Modelling and Evaluation of ATM Networks*, pages 8/1–8/9, July 1994.
25. A.G. Waters. *Multi-party communication over packet networks.* PhD thesis, University of Essex, UK, 1996.
26. B.M. Waxman. Routing of Multipoint Connections. *IEEE journal on selected areas in communications*, 6(9):1617–1622, 1988.
27. L. Wei and D. Estrin. Multicast Routing in Dense and Sparse Modes:Simulation Study of Tradeoffs and Dynamics (95-613). Technical report, Computer Science Department, University of Southern California, 1995.
28. R. Widyono. The Design and Evaluation of Routing Algorithms for Real-time Channels. Tr-94-024, University of California at Berkeley and International Computer Science Institute, September 1994.
29. Q. Zhu, M. Parsa, and J.J. Garcia-Luna-Aceves. A Source-Based Algorithm for Near-Optimum Delay-Constrained Multicasting. In *Proceedings of INFOCOM*, pages 377–385, 1995.