Construction of a proxy-based overlay skeleton tree for large-scale real-time group communications

Author/Contributor:
Guo, Jun; Jha, Sanjay

Construction of a Proxy-based Overlay Skeleton Tree for Large-Scale Real-Time Group Communications

Jun Guo and Sanjay Jha

School of Computer Science and Engineering
The University of New South Wales, Sydney, NSW 2052, Australia
{jguo,sjha}@cse.unsw.edu.au

Abstract. We consider the problem of constructing a proxy-based overlay skeleton tree (POST) in the backbone service domain of a two-tier overlay multicast infrastructure. Spanning all multicast proxies deployed in the overlay backbone, POST acts as an efficient resource sharing platform for supporting large numbers of concurrent multicast sessions, without the need of tree computation for each individual session. The problem is concerned with deciding an appropriate deployment of multicast proxies in the overlay backbone, upon which we wish to find an optimal POST solution so that the maximum end-to-end latency is minimized subject to degree balancing constraints. This problem is shown to be NP-hard. We present a simple heuristic method for deploying multicast proxies, and devise a low complexity greedy algorithm for optimizing the end-to-end latency and degree distribution of POST. Simulation experiments confirm that our proposed approach yields good quality approximate solutions that are close to the optimum.

1   Introduction

Due to the complexity of deploying Internet-wide IP multicast [1] and the inability of end-system multicast to support high-bandwidth delay-sensitive group communications [2, 3], proxy-assisted two-tier overlay multicast has been upheld as a viable alternative to support manageable and scalable inter-domain multicast services for real-time applications on the Internet (see [4–6] and references therein). In this paper, we refer to the proxy-assisted overlay multicast architecture as *overlay multicast* for brevity.

One salient feature of overlay multicast is the set of dedicated multicast proxies that a multicast service provider strategically deploys in the backbone service domain. Advances in hardware technologies have made it possible to endow proxies with large processing power and high fanout capability at ever decreasing cost. Multicast proxies are typically placed at co-location facilities with high-speed connection to the Internet. This way, they can readily utilize the over-provisioned links in the core networks [7] without experiencing significant variations in end-to-end delay [8]. In comparison with peer-to-peer trees formed entirely by end hosts with limited last-mile bandwidth, overlay multicast trees based on multicast proxies are much flatter and wider. Moreover, they

are fairly static and thus eliminate large control overhead caused by dynamic tree maintenance in end-system multicast. Since multicast proxies are application layer servers and thus can use unicast mechanism to communicate between each other, overlay multicast effectively obviates the need for router support as required by IP multicast.

In a high level description, multicast proxies can be classified into edge proxies and transit proxies. As illustrated in Fig. 1, edge proxies are placed at the edge of the overlay backbone. Transit proxies are placed deep in the overlay backbone. Transit proxies and edge proxies constitute Tier 1, while edge proxies and end hosts constitute Tier 2. Each edge proxy acts as a member proxy for the cluster of end hosts within its service area, and sends the multicast traffic into or receives the multicast traffic from the overlay backbone by means of a transit proxy. Each transit proxy is responsible for replicating and forwarding multicast traffic to other transit proxies or edge proxies.

Since overlay multicast is designed to support multiple concurrent multicast sessions in the backbone service domain, multicast proxies essentially constitute a resource sharing environment. This gives rise to the concern of how to manage the multicast state scalability issue, so that the overlay backbone can support as many groups as possible without incurring significant control overhead. In the literature, the idea of aggregated multicast with intergroup tree sharing has been proposed [9]. Multicast sessions with approximately the same set of end hosts are forced to share a single overlay topology to reduce the global control overhead. Based on the aggregated multicast approach, the overlay multicast routing protocol proposed in [6] was shown to significantly reduce the control overhead for establishing and maintaining overlay multicast trees.

In this paper, we introduce the concept of *proxy-based overlay skeleton tree* (POST) and advocate it as an efficient approach to further reduce the computation cost and control overhead of overly multicast trees. POST is computed in such a way that it spans all multicast proxies provisioned in the overlay backbone. The computation of POST is feasible due to the fact that in practice the multicast service provider has complete administrative control over all multicast proxies deployed in the overlay multicast network. POST can be used as a default tree to support any multicast session in the overlay backbone, so long as all branches along the skeleton tree connecting the participating edge proxies have enough bandwidth to support the requested multicast session. For example, the skeleton tree depicted in Fig. 1 can be used to support a multicast session participated by edge proxies B, C and D, so long as all the four branches, namely (B,G), (G,H), (C,H) and (D,H), have enough bandwidth. POST can also be used to support low-overhead control message passing among multicast proxies in the overlay backbone.

Two measures of "goodness" for a POST are end-to-end latency and degree distribution. On the one hand, an overlay routing path joining two edge proxies in the POST is likely to traverse a series of transit proxies. Consequently, it is important to minimize the maximum end-to-end latency (i.e. tree diameter) of the POST. This is essential to the provision of good quality delay-sensitive
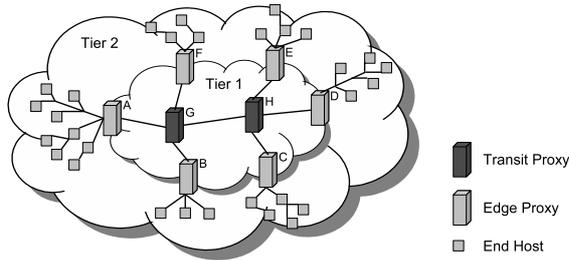
Fig. 1. Proxy-assisted two-tier overlay multicast architecture

multicast services in the overlay multicast network. On the other hand, each transit proxy in the overlay backbone has in general one interface with limited access bandwidth to the Internet. Thus, it is important to distribute the degree evenly between transit proxies for load balancing and to reduce access link stress. Higher link stress indicates greater contention for an interface and is likely to result in higher congestion and packet loss.

Our contribution in this paper is to propose an optimization problem for constructing an efficient POST in the multicast overlay backbone. We also demonstrate in this paper that appropriate placement of multicast proxies can have significant impact on the performance of POST. The optimization problem is concerned with deciding an appropriate deployment of multicast proxies in the overlay backbone, upon which we wish to find an optimal POST solution so that the tree diameter is minimized subject to degree balancing constraints. We frame the problem as a constrained spanning tree problem, which is shown to be NP-hard. We present a simple heuristic method for deploying multicast proxies, and devise a low complexity greedy algorithm for optimizing the end-to-end latency and degree distribution of the POST.

The rest of this paper is organized as follows. We discuss the related work in Sect. 2. Section 3 deals with the formulation of the optimization problem. In Sect. 4, we describe our solution method. Simulation experiments are reported in Sect. 5. Finally, we provide concluding remarks in Sect. 6.

2   Related Work

Existing proposals in [4, 5] for building an overlay multicast backbone tree were all based on the assumption that the proxy deployment is given. Also, these proposals considered tree computation for a single multicast session only. The routing protocols proposed in [4] addressed two constrained spanning tree problems for single-source streaming applications: 1) Minimize the maximum end-to-end latency subject to access bandwidth constraint at each participating multicast proxy; 2) Minimize the average end-to-end latency subject to access bandwidth constraint at each participating multicast proxy. The routing protocols proposed in [5] addressed two constrained spanning tree problems for multi-source multi-

cast applications: 1)Minimize the tree diameter subject to access bandwidth constraint at each participating multicast proxy; 2) Balance the access bandwidth usage at each participating multicast proxy subject to tree diameter bound. The CT algorithm proposed in [5] is closely related to our work, which can be modified to compute a POST in our context.

In [10], Lao, Cui, and Gerla considered the deployment problem for edge proxies in the overlay backbone. They adopted a similar approach to the K-median problem commonly used for web server replica placement in content distribution networks [11]. Their approach is used in this paper for the deployment of edge proxies. In our context, we further study the deployment problem for transit proxies, and demonstrate that it has significant impact on the end-to-end latency performance of POST.

3   Problem Formulation

Consider an overlay network in the form of a complete undirected graph $G = (W, P, E)$. W is the set of N edge proxies which have been deployed using the approach of [10]. P is the set of K potential nodes in the physical topology for deploying M, $M < K$, transit proxies. E is the set of undirected edges, where an edge $(i, j) \in E$ between two nodes $i, j \in W \cup P$ represents the bidirectional unicast path with long-term average latency $l_{i,j}$ between i and j in the physical topology. In practice, unicast latency quantities can be obtained from active or passive measurements of TCP connection round trip time [8, 12].

From P, we wish to decide an appropriate set of nodes P_T for deploying the M transit proxies. We then wish to form a POST, i.e. $T = (W, P_\mathrm{T}, E_\mathrm{T})$, spanning all nodes in $W \cup P_\mathrm{T}$ where E_T are the edges included in T. An edge $(i, j) \in E$ can be included in T if and only if at least one of the two nodes i and j is in P_T. Consequently, the set of internal nodes of T is composed of transit proxies only, and the set of leaf nodes of T is exclusively composed of edge proxies.

For each pair of edge proxies i and j in T, we define $R_{i,j}$ as the set of edges that form the overlay routing path between i and j. Let $L_{i,j}$ denote the latency of the overlay routing path between i and j. Given the unicast latency matrix $\{l_{i,j}\}$, we readily have

$$L_{i,j} = \sum_{(i,j) \in R_{i,j}} l_{i,j} \ . \tag{1}$$

Let L_max denote the maximum end-to-end delay (i.e. the diameter of T), which is given by

$$L_\mathrm{max} = \max_{i<j \in W} L_{i,j} \ . \tag{2}$$

Consider the case where the unicast latency matrix $\{l_{i,j}\}$ satisfies the triangle inequality. Thus, the latency $L_{i,j}$ of the overlay routing path between any two edge proxies i and j can not be smaller than the latency $l_{i,j}$ of the unicast path between the two nodes. This allows us to establish the lower bound (LB) on L_max as

$$L_\mathrm{max}^\mathrm{LB} = \max_{i<j \in W} l_{i,j} \ . \tag{3}$$

Let $d(i)$ denote the degree of transit proxy i. The following proposition states a property on the sum $\sum_{i \in P_T} d(i)$ of degrees of transit proxies in T.

Proposition 1. *The sum of degrees of transit proxies in T is $N + 2M - 2$.*

Proof. Consider the subtree of T composed of M transit proxies only. It follows from Corollary 1.5.3 of [13] that a spanning tree with M nodes has exactly $M-1$ edges. Each such edge is incident to two transit proxies, which contributes two counts towards the sum of degrees of transit proxies. Now, each of the N edge proxies must be connected to one transit proxy, and thus contributes one count towards the degree of the transit proxy to which it is connected. Therefore, the sum of degrees of transit proxies in the entire T is $N + 2M - 2$. $\qquad\square$

We define a degree balancing index F, given by

$$F = \max_{i \in P_T} d(i) - \min_{i \in P_T} d(i) \ . \tag{4}$$

A smaller value of F indicates a more balanced degree distribution among transit proxies. Let F^{LB} denote the lower bound on F. Clearly, $F^{\mathrm{LB}} = 0$ if

$$N + 2M - 2 = kM, \ k = 1, 2, \ldots \tag{5}$$

and we require each transit proxy to have a degree of exactly k in T. In situations where (5) does not hold, $F^{\mathrm{LB}} = 1$. In such cases, letting $k = \lfloor \frac{N+2M-2}{M} \rfloor$ and $n = N + 2M - 2 - kM$, we require n transit proxies to have a degree of exactly $k + 1$, and the remaining $M - n$ transit proxies to have a degree of exactly k.

Before we present the formal statement of the optimization problem, we shall first clarify the motivation of such a problem by considering a 14-node example illustrated in Fig. 2.
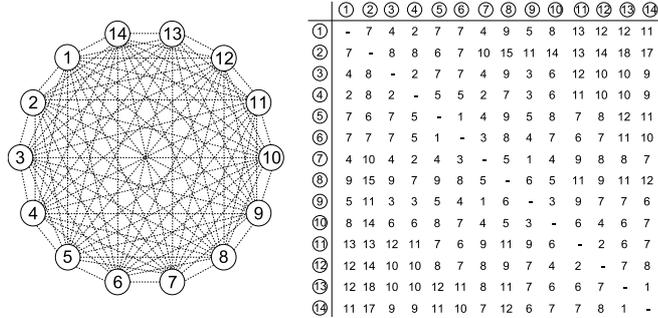

	①	②	③	④	⑤	⑥	⑦	⑧	⑨	⑩	⑪	⑫	⑬	⑭
①	-	7	4	2	7	7	4	9	5	8	13	12	12	11
②	7	-	8	8	6	7	10	15	11	14	13	14	18	17
③	4	8	-	2	7	7	4	9	3	6	12	10	10	9
④	2	8	2	-	5	5	2	7	3	6	11	10	10	9
⑤	7	6	7	5	-	1	4	9	5	8	7	8	12	11
⑥	7	7	7	5	1	-	3	8	4	7	6	7	11	10
⑦	4	10	4	2	4	3	-	5	1	4	9	8	8	7
⑧	9	15	9	7	9	8	5	-	6	5	11	9	11	12
⑨	5	11	3	3	5	4	1	6	-	3	9	7	7	6
⑩	8	14	6	6	8	7	4	5	3	-	6	4	6	7
⑪	13	13	12	11	7	6	9	11	9	6	-	2	6	7
⑫	12	14	10	10	8	7	8	9	7	4	2	-	7	8
⑬	12	18	10	10	12	11	8	11	7	6	6	7	-	1
⑭	11	17	9	9	11	10	7	12	6	7	7	8	1	-

Fig. 2. Overlay topology and unicast latency matrix

We assume that five edge proxies have been deployed at nodes 1, 2, 4, 7 and 10. A routine computation of (3) on the unicast latency matrix $\{l_{i,j}\}$ gives

$L_{\max}^{\mathrm{LB}} = 14$ for this particular instance. From the remaining nine nodes, we wish to choose three nodes for deploying the transit proxies. Since $M = 3$ and $N + 2M - 2 = 9$, we have $F^{\mathrm{LB}} = 0$. To achieve this, we require each transit proxy to have a degree of exactly three in T. An optimal solution displayed in Fig. 3 chooses nodes 5, 6 and 9 for placing the transit proxies, and computes a POST with minimal tree diameter of 14 that hits $L_{\max}^{\mathrm{LB}}$. Our proposed heuristic method chooses nodes 3, 6 and 9 for the transit proxies. The best POST based on this heuristic proxy placement solution achieves $L_{\max} = 18$. In contrast, a random placement of transit proxies at nodes 3, 8 and 12 incurs a large tree diameter of 28. This justifies that appropriate placement of transit proxies in the multicast overlay backbone can have large impact on the latency performance of POST.
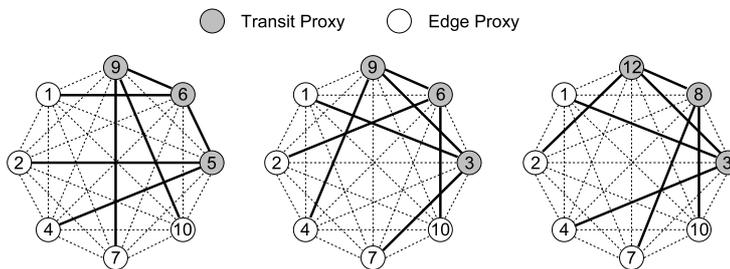

Fig. 3. Solutions for POST

Definition 1. *Minimum diameter degree-balanced spanning tree problem*

Given a complete undirected graph $G = (W, P, E)$, find a constrained spanning tree $T = (W, P_{\mathrm{T}}, E_{\mathrm{T}})$ such that the tree diameter $L_{\max}$ is minimized and T is subject to constraints on: 1) $P_{\mathrm{T}} \in P$, $|P_{\mathrm{T}}| = M$; 2) All nodes in P_{T} are for internal nodes, while all nodes in W are for leaf nodes; 3) F is restricted to F^{LB}.

For brevity, in the rest of this paper, we shall refer to this problem as the MDDB problem. Such a heavily constrained spanning tree problem is NP-hard, since its decision problem can be reduced to the NP-complete weighted diameter problem ([14], page 205). We thus resort to heuristic methods to find near-optimal solutions for this challenging problem.

4   Solution

Our approach for solving the MDDB problem contains two parts. In the first part, we present a simple heuristic method for deploying the set of transit proxies. In the second part, we devise a low complexity algorithm for greedily optimizing the end-to-end latency of T subject to degree balancing among transit proxies.

4.1   Proxy Deployment

For all $i \in P$, we obtain s_i, which is computed by

$$s_i = \sum_{j \in W} l_{i,j} \ .$$

(6)

We choose M nodes in the set P with the smallest values on s_i to constitute the set of transit proxies P_{T}. We define the node with the smallest value on s_i as the *central* proxy, since it has the smallest summed distance to all edge proxies in the set W. For brevity, in the rest of this paper, we refer to this method as HPTP, which stands for Heuristic Placement of Transit Proxies.

4.2   Degree Balancing

An important objective of our greedy algorithm is to guarantee that the degree balancing index F of internal nodes of T is restricted to F^{LB}. As we have discussed in Sect. 3, if $F^{\mathrm{LB}} = 0$, we require each of the M internal nodes to have a degree of exactly k, where $k = \frac{N+2M-2}{M}$. This case can be easily resolved. For each node j that has just been added to the partial tree of T, we check the cumulated degree $d(i)$ of the internal node i which connects node j. If $d(i) = k$, we mark node i so that node i will not be considered by any of the remaining unconnected nodes.

On the other hand, if $F^{\mathrm{LB}} = 1$, we require n internal nodes to have a degree of exactly $k + 1$, and the remaining $M - n$ internal nodes to have a degree of exactly k, where in this case $k = \lfloor \frac{N+2M-2}{M} \rfloor$ and $n = N + 2M - 2 - kM$. To achieve this, we make sure that once the n-th internal node whose degree reaches $k + 1$ has been marked, we further check for each unmarked internal node if its degree has reached k. If so, we mark such nodes accordingly, again to make sure that they will not be considered by any of the remaining unconnected nodes.

4.3   Tree Construction

Let V denote the set of all transit proxies in the partial tree T' of T constructed so far. Let $\overline{V} = P_{\mathrm{T}} - V$ denote the set of all transit proxies not yet in T'. Let $\bar{P}$ denote the set of all internal nodes that have been marked. Let σ_i denote the latency of the longest overlay routing path from an unmarked internal node i to any other node in T'. Starting from the initial T' with V including the central proxy only, for each unconnected transit proxy j in the set $\overline{V}$, we find an unmarked internal node i in T' that minimizes $\delta_j = l_{j,i} + \sigma_i$. We then identify the unconnected transit proxy j in the set $\overline{V}$ with the smallest value on such δ_j, and add it to T' by creating an edge joining node i and node j. After node j is added to the tree, we update V and $\overline{V}$, and we apply the degree balancing module to check if node i needs to be marked into the set $\bar{P}$. This procedure is iterated until all M transit proxies have been connected.

In cases where ties exist either on i or on j, we break the ties on i by choosing node i such that $\sum_{v \in \overline{V}} (l_{v,i} + \sigma_i)$ is the largest. Intuitively, if we do not go for

node i at this iteration, later node i would more likely lead to larger σ_i values of some other transit proxies not yet in the tree. On the other hand, we break the ties on j by choosing node j such that $\sum_{v \in V - \bar{P}}(\sigma_v + l_{v,j})$ is the largest. Similarly, if we do not connect node j at this iteration, later node j would more likely be connected to an internal node which yields larger σ_j value of node j.

After all transit proxies have been connected, we now let V denote the set of all edge proxies in the partial tree T' of T constructed so far, and Let $\overline{V} = W - V$ denote the set of all edge proxies not yet in T'. At any iteration, for each unconnected node j in the set $\overline{V}$, we find an unmarked internal node i in T' which minimizes $\delta_j = l_{j,i} + \sigma_i$. We then identify the unconnected node j in $\overline{V}$ with the largest (not smallest) value on such δ_j, and add it to T' by creating an edge joining node i and node j. Intuitively, if we do not connect node j at this iteration, later it would more likely contribute towards a larger value on the tree diameter. Once node j is added to the tree, we update V and $\overline{V}$, and we again apply the degree balancing module to check if node i needs to be marked into the set $\bar{P}$.

In cases where ties exist either on i or on j, we break the ties on i again by choosing node i such that $\sum_{v \in \overline{V}}(l_{v,i} + \sigma_i)$ is the largest. Intuitively, if we do not go for node i in this case, later node i would certainly lead to larger overlay latency between all nodes it has connected and some unconnected leaf nodes it would connect. On the other hand, we break the ties on j by choosing node j such that $\sum_{v \in P_\mathrm{T} - \bar{P}}(\sigma_v + l_{v,j})$ is the largest. Similarly, if we do not connect node j at this iteration, later node j would more likely be connected to an internal node which contributes towards a larger value on the tree diameter.

It can be shown that this greedy algorithm is $O(MN^2)$ in complexity. For brevity, we refer to it as GOLD (Greedily Optimize Latency and Degree) in the rest of this paper.

5   Simulation Experiments

We have studied the performance of our proposed algorithms for the MDDB problem through detailed simulation experiments. The various network topologies used in our experiments were obtained from the GT-ITM topology generator [15]. We used the flat random graph model for the small size topologies and the transit-stub graph model for the large size topologies. Unicast latency between different pairs of nodes ranges from 1 to 50 ms in the flat random graphs and from 1 to 500 ms in the transit-stub graphs. The number of nodes in the flat random graphs was varied between 14, 16 and 18. All transit-stub graphs had 1,200 nodes. In each graph, edge proxies were placed at a set of nodes, chosen uniformly at random. The number of edge proxies was fixed to five in the flat random graphs and varied between 400, 500 and 600 in the transit-stub graphs. The number of transit proxies was fixed to three in the flat random graphs and varied from 20 to 40 in the transit-stub graphs.

Based on these network topologies, we design the following experiments to study the performance of our proposed algorithms for the MDDB problem. We

have derived an integer linear programming (ILP) formulation for the MDDB problem (see Appendix). The ILP formulation was used to find optimal solutions of the MDDB problem for small size topologies. Note that this ILP model was also used to compute the best POST solution based on a given heuristic proxy placement solution obtained by HPTP. This is done by fixing the indicator variables defined for proxy placement in the ILP model to the corresponding nodes found by HPTP. This allows us to examine the quality of HPTP.

Figure 4 presents the results from experiments with the flat random graphs. For each graph, we have randomly generated 20 sets of different edge proxy placement solutions. We observe from these results that the performance of HPTP is reasonably good, considering that it is such a simple approach. In many cases, HPTP hits the optimal solution identified by ILP. The worst case performance among these results is 40% away from the optimum. Since we are not able to use ILP to solve for large size problem instances due to its known exponential computational complexity, we used GOLD to compute approximate POST solutions for HPTP on large size topologies.

The CT algorithm proposed in [5] was designed for a minimum diameter degree-limited spanning tree problem to address efficient routing in the multicast overlay backbone. It is similar to Prim's algorithm [16] for minimum spanning tree in a sense that the tree construction process starts from an arbitrarily selected node and takes an arbitrary tie-breaking approach. In contrast, GOLD starts from the central proxy of the overlay backbone and takes an elaborate tie-breaking approach in the tree construction process. For the purpose of comparison, we have directly combined the CT algorithm with the degree balancing module of GOLD. The resulting tree construction method is referred to as enhanced CT (ECT) in the simulation experiments.

Given that we have established the lower bound on $L_{\max}$, we used $L_{\max}^{\mathrm{LB}}$ as the benchmark to measure the quality of HPTP on large size topologies. The $L_{\max}$ results presented in Fig. 5 were plotted in the form of percentage deviation from $L_{\max}^{\mathrm{LB}}$, given by $(L_{\max} - L_{\max}^{\mathrm{LB}})/L_{\max}^{\mathrm{LB}}(\%)$. These results clearly demonstrate the enhanced performance of GOLD in comparison with ECT. In all cases, GOLD improves $L_{\max}$ by up to 4.4%. Such a performance gain is only at the expense of a modicum of additional computation cost, as demonstrated in Table 1. Moreover, HPTP shows sufficiently good performance on these transit-stub graphs. Even though we rely on GOLD to find approximate POST solutions, the percentage deviation is yet less than 18% in all cases.

Table 1. CPU time on a 3.2GHz Xeon machine for $M = 40$

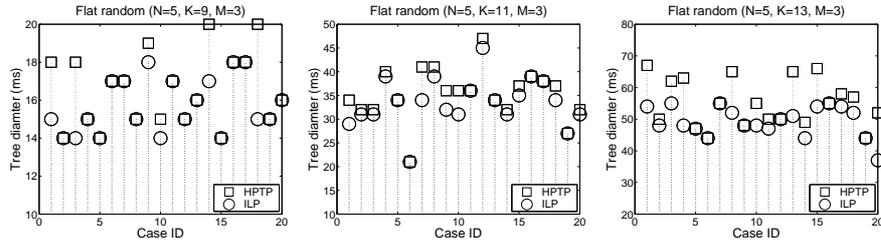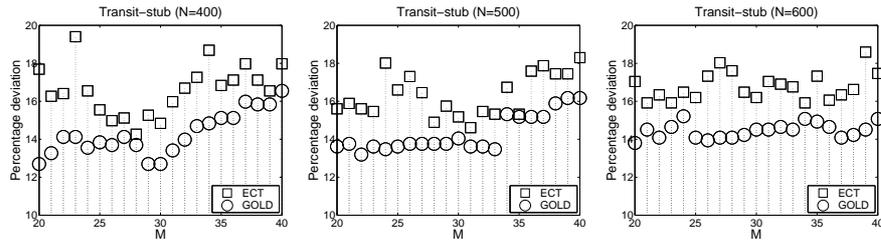|	$N = 400$	$N = 500$	$N = 600$
ECT	0.84 sec	1.27 sec	1.80 sec
GOLD	0.90 sec	1.35 sec	1.95 sec

Fig. 4. Quality of HPTP

Fig. 5. Performance comparison between ECT and GOLD

6   Conclusion and Future Work

In this paper, we have introduced the concept of POST and advocated it as an efficient approach to support large-scale real-time group communications in two-tier overlay multicast networks. We have demonstrated that appropriate placement of multicast proxies can have significant impact on the latency performance of POST. We have proposed a constrained spanning tree problem to find optimal solutions for POST such that its maximum end-to-end latency can be minimized subject to degree balancing constraints. A low complexity heuristic approach has been developed, which has been shown by simulation experiments to be scalable for large-size overlay multicast networks and yield good quality solutions of POST. The proposed algorithm can be utilized by a weight-coded genetic algorithm to further improve the solution quality [17]. In our future work, we plan to design efficient restoration algorithms to endow POST with resilience capability [18].

References

1. Diot, C., Levine, B.N., Lyles, B., Kassem, H., Balensiefen, D.: Deployment issues for the IP multicast service and architecture. IEEE Network **14**(1) (Jan./Feb.

2000) 78–88

2. Banerjee, S., Bhattacharjee, B., Kommareddy, C.: Scalable application layer multicast. In: Proc. ACM SIGCOMM 02. (Aug. 2002) 205–217
3. Chu, Y.H., Rao, S.G., Zhang, H.: A case for end system multicast. In: Proc. ACM SIGMETRICS 00. (Jun. 2000) 1–12
4. Banerjee, S., Kommareddy, C., Kar, K., Bhattacharjee, B., Khuller, S.: Construction of an efficient overlay multicast infrastructure for real-time applications. In: Proc. IEEE INFOCOM 03. (Mar. 2003) 1521–1531
5. Shi, S.Y., Turner, J.S.: Routing in overlay multicast networks. In: Proc. IEEE INFOCOM 02. (Jun. 2002) 1200–1208
6. Lao, L., Cui, J.H., Gerla, M.: TOMA: a viable solution for large-scale multicast service support. In: Proc. IFIP NETWORKING 05. (May 2005) 906–917
7. Bhattacharyya, S., Diot, C., Jetcheva, J., Taft, N.: Pop-level and access-link-level traffic dynamics in a tier-1 POP. In: Proc. ACM IMW 01. (Nov. 2001) 39–53
8. Jaiswal, S., Iannaccone, G., Diot, C., Kurose, J., Towsley, D.: Inferring TCP connection characteristics through passive measurements. In: Proc. IEEE INFOCOM 04. (Mar. 2004) 1582–1592
9. Fei, A., Cui, J., Gerla, M., Faloutsos, M.: Aggregated multicast: an approach to reduce multicast state. In: Proc. IEEE GLOBECOM 01. (Nov. 2001) 1595–1599
10. Lao, L., Cui, J.H., Gerla, M.: Multicast service overlay design. In: Proc. SPECTS 05. (Jul. 2005)
11. Qiu, L., Padmanabhan, V.N., Voelker, G.M.: On the placement of web server replicas. In: Proc. IEEE INFOCOM 01. (Apr. 2001) 1587–1596
12. Paxson, V.: End-to-end internet packet dynamics. IEEE/ACM Trans. Networking **7**(3) (Jun. 1999) 277–292
13. Diestel, R.: Graph Theory. 3rd edn. Springer-Verlag, New York (2005)
14. Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness. W. H. Freeman, San Francisco (1979)
15. Zegura, E.W., Calvert, K.L., Bhattacharjee, S.: How to model an internetwork. In: Proc. IEEE INFOCOM 96. (Mar. 1996) 594–602
16. Cormen, T.H., Leiserson, C.E., Rivest, R.L., Stein, C.: Introduction to Algorithms. 2nd edn. MIT Press, Cambridge, M.A. (2001)
17. Guo, J., Jha, S., Banerjee, S.: GOLD: An overlay multicast tree with globally optimized latency and out-degree. Technical Report UNSW-CSE-TR-0615, The University of New South Wales, Australia (2006)
18. Lau, W., Jha, S., Banerjee, S.: Efficient bandwidth guaranteed restoration algorithms for multicast connections. In: Proc. IFIP NETWORKING 05. (May 2005) 1005–1017

Appendix

Here we provide an ILP formulation for the MDDB problem. For the convenience of forming the overlay routing path between any two edge proxies in the ILP formulation, we define E' as a set of directed edges that include both directed edges $\langle i,j \rangle$ and $\langle j,i \rangle$ for each undirected edge (i,j) in E. Let the 0-1 variables x_i, $i \in P$, indicate if a transit proxy is deployed at node i. Let the 0-1 variables $p_{i,j}^{(m,n)}$, $m < n \in W$, $\langle i,j \rangle \in E'$, indicate if the directed edge $\langle i,j \rangle$ is included in the overlay routing path between node m and node n. Let the 0-1 variables $t_{i,j}$,

$(i, j) \in E$, indicate if the undirected edge (i, j) is used by the POST. Let the non-negative variables $d_{\max}$ and $d_{\min}$ respectively identify the maximum degree and the minimum degree among the transit proxies. For the latter, it is useful to define a non-negative variable y_i for each $i \in P$. Let the non-negative variable z identify the diameter of the POST.

The MDDB problem can now be formulated as:

$$\text{Minimize} \quad z \tag{7}$$

subject to

$$\sum_{i \in P} x_i = M \tag{8}$$

$$\sum_{j:\langle i,j \rangle \in E'} p_{i,j}^{(m,n)} - \sum_{j:\langle j,i \rangle \in E'} p_{j,i}^{(m,n)} = \begin{cases} 1, & \text{if } i = m \\ 0, & \text{if } i \in W \cup P - \{m, n\} \\ -1, & \text{if } i = n \end{cases}$$
$$\forall m < n \in W \tag{9}$$

$$\sum_{m < n \in W} \left[p_{i,j}^{(m,n)} + p_{j,i}^{(m,n)} \right] \leq N \cdot (N-1) \cdot t_{i,j}, \ \forall (i,j) \in E \tag{10}$$

$$\sum_{(i,j) \in E} t_{i,j} = N + M - 1 \tag{11}$$

$$\sum_{j:(i,j) \in E} t_{i,j} + \sum_{j:(j,i) \in E} t_{j,i} \leq 1, \ \forall i \in W \tag{12}$$

$$\sum_{j:(i,j) \in E} t_{i,j} + \sum_{j:(j,i) \in E} t_{j,i} \leq (N + M - 1) \cdot x_i, \ \forall i \in P \tag{13}$$

$$z \geq \sum_{\langle i,j \rangle \in E'} p_{i,j}^{(m,n)} l_{i,j}, \ \forall m < n \in W \tag{14}$$

$$d_{\max} \geq \sum_{j:(i,j) \in E} t_{i,j} + \sum_{j:(j,i) \in E} t_{j,i}, \ \forall i \in P \tag{15}$$

$$y_i = (N + M - 1) \cdot (1 - x_i) + \sum_{j:(i,j) \in E} t_{i,j} + \sum_{j:(j,i) \in E} t_{j,i}, \ \forall i \in P \tag{16}$$

$$d_{\min} \leq y_i, \ \forall i \in P \tag{17}$$

$$d_{\max} - d_{\min} \leq F^{\mathrm{LB}} \tag{18}$$

Equation (8) restricts that M nodes in P are selected as transit proxies. Equations (9) and (10) ensure that the solution is a spanning tree that includes all edge proxies. More explicitly, they enforce one single overlay routing path between any two edge proxies. Equations (11) to (13) ensure that the set of leaf nodes of the spanning tree is exclusively composed of edge proxies, and the set of internal nodes is composed of transit proxies only. Equation (14) identifies the diameter of the spanning tree. Equations (15) to (18) enforce the balance of degree among transit proxies. All the equations jointly ensure that the solution is a POST that satisfies all constraints of the MDDB problem.