A Distributed Algorithm for Overlay Backbone Multicast Routing in Content Delivery Networks

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. To support large-scale live Internet broadcasting services efficiently in content delivery networks (CDNs), it is essential to exploit peer-to-peer capabilities among end users. This way, the access bandwidth demand on CDN servers in the overlay backbone can be largely reduced. Such a streaming infrastructure gives rise to a challenging overlay backbone multicast routing problem (OBMRP) to optimize multicast routing among CDN servers in the overlay backbone. In this paper, we take a graph theoretic approach and frame OBMRP as a constrained spanning tree problem which is shown to be NP-hard. We present a lightweight distributed algorithm for OBMRP. Simulation experiments confirm that our proposed algorithm converges to good quality solutions and requires small control overhead.

1 Introduction

Recent experiences of Internet service providers have seen a huge market demand for live Internet broadcasting services in content delivery networks (CDNs) [1]. AOL's airing of the Live 8 concerts in July 2005 drew about 5 million users within one day and delivered 175,000 concurrent video streams of the concerts at its peak. This record was soon broken by Yahoo!'s broadcasting of NASA's Shuttle Discovery launch, which is said to have delivered more than 335,000 video streams simultaneously. MSN's latest broadcasting of the Live Earth concerts in July 2007, using Akamai's streaming platform [2], has also drawn a global audience of the order of millions of viewers across the Internet.

For such an Internet killer application that is both resource-intensive and latency-sensitive, a challenging issue is to design bandwidth-efficient mechanisms and algorithms that can handle large-scale live Internet broadcasting of streaming video efficiently in CDNs. While IP multicast [3] is doubtlessly the ideal solution for supporting large-scale live Internet broadcasting services, enabling IP multicast across the global Internet has not been successful due to its various deployment issues [4]. Until the global deployment of IP multicast could be eventually realized, Internet service providers have to resort to interim solutions. Among them, current service providers in general favor the overlay multicast approach that emulates the forwarding mechanism of IP multicast at the application layer [5,6,7,8,9,10,11,12,13].

© IFIP International Federation for Information Processing 2008

In particular, small-scale service providers (such as PPLive [14]) typically use the peer-to-peer (P2P) approach [5,6,7,8]. In the P2P approach, end users wishing to participate in a live broadcasting session self-organize into an overlay multicast tree rooted at the origin server. The P2P approach, however, cannot yield satisfactory performance in large-scale live broadcasting services [14]. This is mainly due to the fact that current broadband access technologies provide end users with very limited upload bandwidth. The fanout capability of end users is thus significantly restricted. As a result, overlay multicast trees due to the P2P approach could be rather tall, so that end users deep in the tree (hence far from the origin server) are likely to suffer considerable lag [14].

On the other hand, large-scale service providers (such as Akamai [2]) generally adopt the infrastructure-based approach [9,10,11,12,13]. This approach relies on a set of geographically distributed and dedicated servers with large processing power and high fanout capability. Such powerful servers are typically placed at co-location facilities with high-speed connection to the Internet, and thus form a backbone service domain, which we call *overlay backbone*, for the overlay multicast network. The infrastructure-based approach has been widely used by commercial CDNs to support live Internet broadcasting services [2]. Akamai's streaming platform largely benefits from a CDN that consists of over 20,000 servers distributed in more than 70 countries. Its solution is, however, a pure infrastructure-based approach, where each end user wishing to view the live broadcasting event is directly connected to one CDN server in the overlay backbone to fetch the live video stream [2].

The pure infrastructure-based approach is not efficient in handling large-scale live Internet broadcasting services, since it is less effective in restricting excessive and redundant traffic to be injected into the Internet, and also because the tremendous access bandwidth demand on CDN servers would render large-scale Internet broadcasting services sheer prohibitive. One flexible and scalable approach is to exploit P2P capabilities among end users as much as possible, so long as a predefined bound on the end-to-end latency can be met [9]. Such a hybrid approach leads to an effective two-tier overlay multicast architecture [13], in which the access bandwidth demand on CDN servers in the overlay backbone can be largely reduced.

The two-tier overlay multicast architecture gives rise to a challenging overlay multicast problem among CDN servers in the overlay backbone. For a live broadcasting event, each participating CDN server is made aware of the largest delay from it to end users within its service area as well as the number of end users within its service area. The problem is to optimize the overlay multicast tree among CDN servers in the overlay backbone subject to access bandwidth constraints on CDN servers and the fixed bound on the de facto maximal end-to-end latency from the origin server to end users. The optimization criterion is to minimize the weighted average latency from the origin server to the participating CDN servers, which weights each CDN server with the number of end users within its service area. We call such a problem as the *overlay backbone multicast routing problem* (OBMRP) in this paper. OBMRP is strongly motivated due to the concern that the larger the size of user population within the service area of a CDN server, the larger is the access bandwidth demand that could be placed on the corresponding server. By reducing the latency as much as possible at such servers, it is more feasible to exploit P2P capabilities among end users and thus reduces the access bandwidth demand on CDN servers.

In this paper, we take a graph theoretic approach to frame OBMRP as a constrained spanning tree problem and show that it is NP-hard. We propose a lightweight distributed algorithm for OBMRP (hence called OBMRP-DA for brevity). For the purpose of validating the efficacy of OBMRP-DA, we modify the distributed algorithm proposed in [11] for the OMNI problem (hence called OMNI-DA for brevity) to provide a comparable solution for OBMRP.

In OMNI, each CDN server in the overlay backbone is made aware of only the number of end users within its service area. The overlay multicast problem addressed by OMNI-DA is to minimize the weighted average latency from the origin server to the participating CDN servers subject to access bandwidth constraints on CDN servers. OMNI-DA cannot guarantee to bound the de facto maximal end-to-end latency as required in the context of OBMRP. Moreover, OMNI-DA relies on a set of local transformation operations which essentially require probing among all nodes within up to two levels of each other. It was observed that such local transformation operations cannot guarantee that a global minimum will be achieved. OMNI-DA thus relies on a random swap operation to divert the algorithm from the local minimum region. Such a random swap operation inevitably introduces significant oscillations to the tree latency and thus prolongs the converging process. We shall see in this paper that our proposed OBMRP-DA benefits from the more efficient transformation operations that we have designed, and thus can significantly reduce the control overhead while yielding solutions of comparable qualities to OMNI-DA.

The rest of this paper is organized as follows. Section 2 deals with the problem formulation. Section 3 presents the distributed algorithm. Simulation experiments are reported in Sect. 4. Finally, we draw conclusions in Sect. 5.

2 Problem Formulation

We model the overlay backbone in the CDN as a complete directed graph $G = (V, E)$, where V is the set of N nodes and $E = V \times V$ is the set of edges. Each node in V represents a CDN server participating in the live broadcasting session. Let node r be the origin server. All other nodes (in the set $V - \{r\}$) are proxy servers. The directed edge $\langle i, j \rangle$ in E represents the unicast path of latency $l_{i,j}$ from node i to node j. By $\{l_{i,j}\}$, we denote the matrix of unicast latency quantities between each pair of nodes in G.

An overlay backbone multicast tree can be represented by a directed spanning tree T of G rooted at node r. For each sink node in the set $V - \{r\}$, we define $R_{r,v}$ as the set of directed edges that form the overlay routing path from the root node to node v in the multicast tree. Let $L_{r,v}$ denote the aggregate latency along the overlay routing path from the root node to node v. Let γ_v be the largest delay from node v to end users within its service area, and c_v be the number of end users within the service area of node v. The weight w_v of node v is given by

$$w_v = c_v / \sum_{v \in V - \{r\}} c_v . \tag{1}$$

Let $\bar{L}$ denote the weighted sum of latencies from the origin server to the proxy servers along T. Let $L_{\max}$ denote the de facto maximal end-to-end latency from the origin server to the end users. Let $L_{\max}^{\mathrm{B}}$ denote the specified bound on $L_{\max}$. Given the unicast latency matrix $\{l_{i,j}\}$, we readily have

$$L_{r,v} = \sum_{\langle i,j \rangle \in R_{r,v}} l_{i,j} \tag{2}$$

and

$$\bar{L} = \sum_{v \in V - \{r\}} w_v L_{r,v} \tag{3}$$

and

$$L_{\max} = \max_{v \in V - \{r\}} \left(L_{r,v} + \gamma_v \right) . \tag{4}$$

Let $\widetilde{d_i}$ be the residual degree of node i, and d_i be the out-degree of node i counted within the overlay backbone multicast tree. Since a spanning tree with N nodes has exactly $N-1$ edges, the sum of out-degrees in the overlay backbone multicast tree satisfies $\sum_{i \in V} d_i = N - 1$.

Definition 1. *Given the complete directed graph $G = (V, E)$, OBMRP is to find a constrained directed spanning tree T of G rooted at node r, such that $\bar{L}$ is minimized, while T satisfies the residual degree constraint, i.e. $d_i \leq \widetilde{d_i}$, $\forall i \in V$, and the bound on the de facto maximal end-to-end latency, i.e. $L_{\max} \leq L_{\max}^{\mathrm{B}}$.*

Such a constrained spanning tree problem is NP-hard. This is shown by creating a dummy node for each node v in $V - \{r\}$ and forming an edge of weight γ_v between node v and its corresponding dummy node. The resulting problem can be reduced to finding a Hamiltonian path within the augmented graph which is known to be NP-complete [15].

3 Distributed Algorithm

In this paper, we define the *ancestor nodes* of node i as those nodes (including the root node) in the overlay routing path of the multicast tree from the root node to node i, where the *parent node* of node i is the immediate forwarding node to node i in the overlay routing path. The *child nodes* of node i are the immediate nodes that relay from node i in the subtree rooted at node i. The *descendants* of node i are defined as those end users served by proxy servers (including node i) in the subtree rooted at node i.

Our proposed distributed algorithm for OBMRP requires node i, $\forall i \in V$, to maintain the following state information during the iterative tree restructuring process, except for those inapplicable to the root node:

- $\hat{i}$: Parent node of node i in the current tree
- Φ_i: Set of all ancestor nodes of node i in the current tree
- Ω_i: Set of all child nodes of node i in the current tree
- $\widetilde{d_i}$: Residual degree of node i in the current tree
- W_i: Aggregate weight of all nodes in the subtree rooted at node i (including node i), which is given by

$$W_i = \begin{cases} w_i, \text{ if node } i \text{ is a leaf node} \\ w_i + \sum\limits_{j \in \Omega_i} W_j, \text{ elsewhere} \end{cases} \tag{5}$$

- $L_{r,i}$: Latency from the root node to node i, which is given by

$$L_{r,i} = L_{r,\hat{i}} + l_{\hat{i},i} \tag{6}$$

- Γ_i: Largest delay from node i to end users served by proxy servers (including node i) in the subtree rooted at node i. This can be iteratively computed by

$$\Gamma_i = \begin{cases} \gamma_i, \text{ if node } i \text{ is a leaf node} \\ \max\left[\max\limits_{j \in \Omega_i}(l_{i,j} + \Gamma_j), \gamma_i\right], \text{ elsewhere} \end{cases} \tag{7}$$

Clearly, Γ_r corresponds to $L_{\max}$ of the current tree.

3.1 Tree Initialization

Starting from the initial tree including the origin server only, we allow proxy servers in $V - \{r\}$ to join the tree in a random order, which is realistic in an actual live broadcasting event. If multiple nodes arrive concurrently, they are added to the tree in the decreasing order according to their node IDs. When node i wishes to join the tree, it directly contacts the root node, since the root node by default maintains the list of all nodes in the current tree. Upon the joining request of node i, the root node successively probes the list of nodes in the current tree until node j with sufficient residual degree, i.e. $\widetilde{d_j} > 0$, is found. Node i is thus grafted to the tree by registering as a child node of node j.

3.2 Handling of Tree Restructuring Requests

Our proposed distributed algorithm for OBMRP works by allowing each node in T to periodically and randomly contact another node in T to make a tree restructuring request. Let $\overrightarrow{i,j}$ denote such a tree restructuring request from node i to node j. Following OBMRP-DA, $\overrightarrow{i,j}$ will be simply rejected by node j if the current states and positions of node i and node j in T satisfy one of the five exclusive conditions: 1) $i \in \Phi_j$; 2) $j = r$, $j = \hat{i}$; 3) $j = r$, $j \neq \hat{i}$, $\widetilde{d_j} = 0$; 4) $j \neq r$, $j = \hat{i}$, $\widetilde{d_{\hat{j}}} = 0$; 5)$j \in \Phi_i - \{r\}$, $j \neq \hat{i}$, $\widetilde{d_{\hat{j}}} = 0$, $\widetilde{d_j} = 0$. Node j considers $\overrightarrow{i,j}$ only if the current states and positions of node i and node j in T match one of the five distinct cases illustrated in Fig. 1. In particular, (a), (b) and (c) deal with the various cases where we have $j \in \Phi_i$.

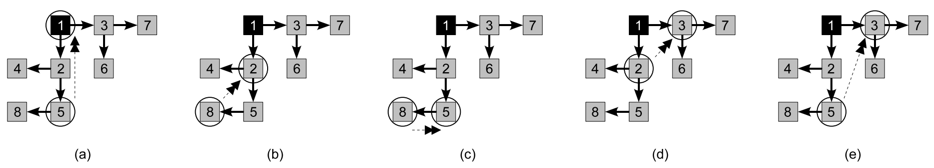

Fig. 1. Snapshot of the five distinct cases where a tree restructuring request from node i will be considered by node j. Both node i and node j are marked by circles. Node j is pointed by the double arrow.

(a) If $j \neq r$, $j = \hat{i}$ and $\widetilde{d}_{\hat{j}} > 0$, a Type-A transformation (described in Sect. 3.3) will be attempted by node j.
(b) If $j = r$, $j \neq \hat{i}$ and $\widetilde{d}_j > 0$, a Type-C transformation (described in Sect. 3.5) will be attempted by node j.
(c) If $j \in \Phi_i - \{r\}$, $j \neq \hat{i}$ and $\widetilde{d}_{\hat{j}} > 0$, a Type-A transformation will be attempted by node j. If $\widetilde{d}_{\hat{j}} = 0$ but $\widetilde{d}_j > 0$, node j will instead attempt a Type-C transformation.
(d) If $\hat{j} = \hat{i}$, a Type-C transformation will be attempted by node j provided $\widetilde{d}_j > 0$. However, if $\widetilde{d}_j = 0$, node j will instead attempt a Type-D transformation (described in Sect. 3.6).
(e) In all other situations where neither $i \in \Phi_j$ nor $j \in \Phi_i$, node j will selectively attempt a Type-A, B (described in Sect. 3.4), C or D transformation (in the corresponding order), depending on if any of the four exclusive conditions, i.e. $\widetilde{d}_{\hat{j}} > 0$, $\widetilde{d}_{\hat{j}} = 0$, $\widetilde{d}_j > 0$, or $\widetilde{d}_j = 0$, holds true.

3.3 Type-A Transformation

Since $\widetilde{d}_{\hat{j}} > 0$, node i (together with its subtree if any) is switched to be a child node of node $\hat{j}$, given that

$$\breve{L}_{r,i} = (L_{r,\hat{i}} + l_{\hat{i},i}) - (L_{r,\hat{j}} + l_{\hat{j},i}) > 0 \tag{8}$$

where $\breve{L}_{r,i}$ denotes the amount of variation on $L_{r,i}$. Clearly, such a transformation reduces the end-to-end latency $L_{r,j}$ of each node j in the subtree rooted at node i by $\breve{L}_{r,i}$, and thus reduces $\bar{L}$ by $W_i \cdot \breve{L}_{r,i}$ without increasing Γ_r of the current tree. We illustrate an example of the Type-A transformation in Fig. 2(a) for the case presented in Fig. 1(e).

3.4 Type-B Transformation

Since $\widetilde{d}_{\hat{j}} = 0$, if (8) holds true, we may switch node j to be either a child node of node $\hat{i}$, or a child node of node i after node i is grafted to node $\hat{j}$. More explicitly, we check if

$$\breve{L}'_{r,j} = (L_{r,\hat{j}} + l_{\hat{j},j}) - (L_{r,\hat{i}} + l_{\hat{i},j}) > 0 \tag{9}$$

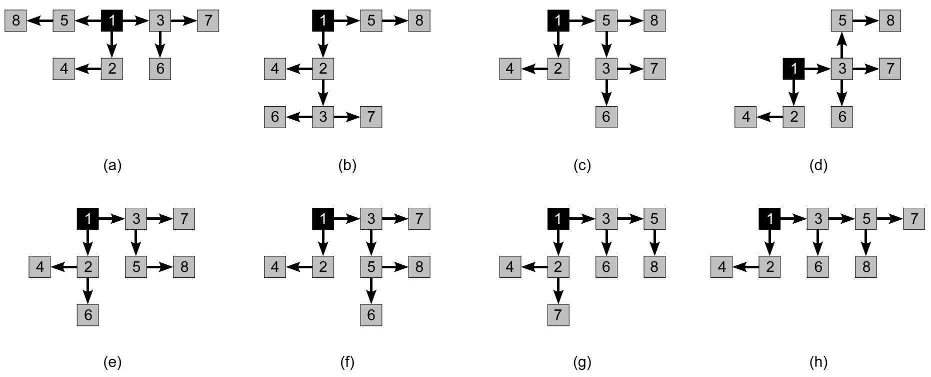

Fig. 2. All possible tree restructuring results for the case depicted in (e) of Fig. 1: (a) due to the Type-A transformation; (b) and (c) due to the Type-B transformation; (d) due to the Type-C transformation; (e) to (h) due to the Type-D transformation.

and if

$$\breve{L}''_{r,j} = (L_{r,\hat{\jmath}} + l_{\hat{\jmath},j}) - (L_{r,\hat{\jmath}} + l_{\hat{\jmath},i} + l_{i,j}) > 0 \tag{10}$$

given that $\widetilde{d}_i > 0$. Let $\breve{L}_{r,j} = \max(\breve{L}'_{r,j}, \breve{L}''_{r,j})$. If $\breve{L}_{r,j} > 0$, we perform the transformation by choosing the option that yields $\breve{L}_{r,j}$. On the other hand, if $\breve{L}_{r,j} \leq 0$, but we have

$$W_i \cdot \breve{L}_{r,i} + W_j \cdot \breve{L}_{r,j} > 0, \tag{11}$$

we proceed with the transformation by choosing the option that achieves $\breve{L}_{r,j}$. However, this is done only if the transformation does not increase Γ_r of the current tree or violate $L^{\mathrm{B}}_{\max}$ if Γ_r is already smaller than $L^{\mathrm{B}}_{\max}$. The two possible tree updates due to the Type-B transformation of Fig. 1(e) are shown in Fig. 2(b) and Fig. 2(c).

3.5 Type-C Transformation

This transformation is similar to the Type-A transformation by instead checking if

$$\breve{L}_{r,i} = (L_{r,\hat{\imath}} + l_{\hat{\imath},i}) - (L_{r,j} + l_{j,i}) > 0 \tag{12}$$

If so, node i (together with its subtree if any) is simply grafted to node j, since such a transformation certainly reduces $\bar{L}$ by $W_i \cdot \breve{L}_{r,i}$ without increasing Γ_r. An example of the Type-C transformation of Fig. 1(e) is shown in Fig. 2(d).

3.6 Type-D Transformation

Given that $\widetilde{d}_j = 0$ in this case, if (12) holds true, for each child node c of node j, i.e. $c \in \Omega_j$, we may switch node c to be either a child node of node $\hat{\imath}$, or a child node of node i after node i is grafted to node j. More explicitly, we check for all $c \in \Omega_j$ if

$$\breve{L}'_{r,c} = (L_{r,j} + l_{j,c}) - (L_{r,\hat{i}} + l_{\hat{i},c}) > 0 \tag{13}$$

and if

$$\breve{L}''_{r,c} = (L_{r,j} + l_{j,c}) - (L_{r,j} + l_{j,i} + l_{i,c}) > 0 \tag{14}$$

given that $\widetilde{d_i} > 0$. Let $\breve{L}_{r,c} = \max(\breve{L}'_{r,c}, \breve{L}''_{r,c})$, and let c' denote the node in Ω_j such that

$$W_{c'} \cdot \breve{L}_{r,c'} = \max_{c \in \Omega_j} W_c \cdot \breve{L}_{r,c}. \tag{15}$$

If $\breve{L}_{r,c'} > 0$, we perform the transformation by choosing the option that yields $\breve{L}_{r,c'}$. On the other hand, if $\breve{L}_{r,c'} \leq 0$, but we have

$$W_i \cdot \breve{L}_{r,i} + W_{c'} \cdot \breve{L}_{r,c'} > 0, \tag{16}$$

we proceed with the transformation by choosing the option that achieves $\breve{L}_{r,c'}$. However, this is done only if the transformation does not increase Γ_r of the current tree or violate $L^{\mathrm{B}}_{\max}$ if Γ_r is already smaller than $L^{\mathrm{B}}_{\max}$. All four possible tree updates due to the Type-D transformation of Fig. 1(e) are illustrated in Fig. 2(e) to Fig. 2(h), respectively.

3.7 Control Overhead

The control overhead of OBMRP-DA is small due to the four lightweight transformation operations that we design. For a tree restructuring request $\overrightarrow{i,j}$, the Type-A transformation only requires probing between node i and node $\hat{j}$ ($i \leftrightarrow \hat{j}$ for short) to acquire the unicast latency quantity $l_{\hat{j},i}$. Similarly, the Type-C transformation only requires probing between $i \leftrightarrow j$ to acquire the unicast latency quantity $l_{j,i}$. The Type-B transformation requires probing between three pairs of nodes, i.e. $i \leftrightarrow \hat{j}$, $\hat{i} \leftrightarrow j$ and $i \leftrightarrow j$, to acquire the corresponding unicast latency quantities. Even the most elaborate Type-D transformation merely requires probing between $i \leftrightarrow j$, $\hat{i} \leftrightarrow c$ and $i \leftrightarrow c$, $\forall c \in \Omega_j$, which is within only one level of node j and thus requires only $O(|\Omega_j|)$ control overhead.

It is also important to note that OBMRP-DA defines exclusive conditions for an appropriate transformation operation to be performed, based on the current states and positions of node i and node j in the multicast tree. Consequently, any successful tree restructuring request $\overrightarrow{i,j}$ requires no more than one transformation attempt. The time for which the multicast tree is left in a transient state due to the transformation operation is thus minimal.

4 Simulation Experiments

We have examined the efficacy of our proposed OBMRP-DA through extensive simulation experiments. The network topologies used in our experiments are obtained from the transit-stub graph model of the GT-ITM topology generator [16]. All topologies have 12,000 nodes. CDN servers are placed at a set of N nodes, chosen uniformly at random. Due to the space limitation, here we report experiments on one small-size graph ($N = 20$) and ten large-size graphs ($N = 300$). Unicast latencies between different pairs of nodes in these overlay topologies vary from 10 to 1000 msec. We set the residual degree as 5 for each node in the small-size graph and 10 in the large-size graphs. The bound $L_{\max}^{\mathrm{B}}$ is set to 2000 in all experiments. The number c_i of node i is randomly selected between 100 and 600. The value γ_i of node i is randomly generated following a lognormal distribution Log-N(a, b) with $a = c_i$ and $b = c_i$. We thus introduce certain statistical correlation between c_i and γ_i. This is due to the concern that it is likely that a larger number of end users within the service area of a CDN server can cause a larger delay from the server to its end users.

Based on these overlay topologies, we design the following experiments to study the performance of OBMRP-DA. We have derived an integer linear programming (ILP) formulation for OBMRP (OBMRP-ILP for brevity, provided in the Appendix). Although ILP is known to have an exponential computational complexity and is unlikely to be solved for large-size problem instances, it allows us to find optimal solutions for small-size problem instances of OBMRP. Since OBMRP is NP-hard and the proposed OBMRP-DA uses a random approach for tree initialization, independent runs of OBMRP-DA for the same experiment setting can converge to different solutions. For each experiment setting, we thus solve OBMRP-DA for 200 independent runs, each of which for up to $300 \times N$ successive tree restructuring requests.

Results in Fig. 3(a) for the small-size graph are presented in the form of percentage deviation in $\bar{L}$ between the mean results obtained from OBMRP-DA and the optimal solutions from OBMRP-ILP. Each data point in Fig. 3(a) corresponds to the case where we set a particular node in the graph with the indicated node ID to be the root node. We see in all instances that the solutions obtained from OBMRP-DA are very close to the optimal solutions. The worst case performance in $\bar{L}$ is within only 6% of OBMRP-ILP.

As we have discussed in Sect. 1, OMNI-DA proposed in [11] can be modified to provide a comparable solution for OBMRP. For the purpose of comparison, we have modified OMNI-DA by adding the same elements that we have developed for OBMRP-DA to control Γ_r during the tree restructuring process. While the random swap operation defined in OMNI-DA can have the same effect in improving the latency performance in the context of OBMRP-DA, we did not enable it in our experiments so as to make a fair comparison between the deterministic transformation operations proposed in this paper and those in [11].

For each of the ten large-size graphs, we again set each particular node to be the root node. This creates 3,000 simulation experiments. In each experiment, we compare the mean $\bar{L}$ performance between OBMRP-DA and OMNI-DA in the form of a ratio. We also count the number of probing actions required in the tree restructuring process. This is for us to investigate and compare the control overhead (again in the form of a ratio) between the two different algorithms. The 3,000 data points obtained from the $\bar{L}$ comparison and those from the control overhead comparison are presented in Fig. 3(b) and Fig. 3(c), respectively, both in the form of cumulative percentage.

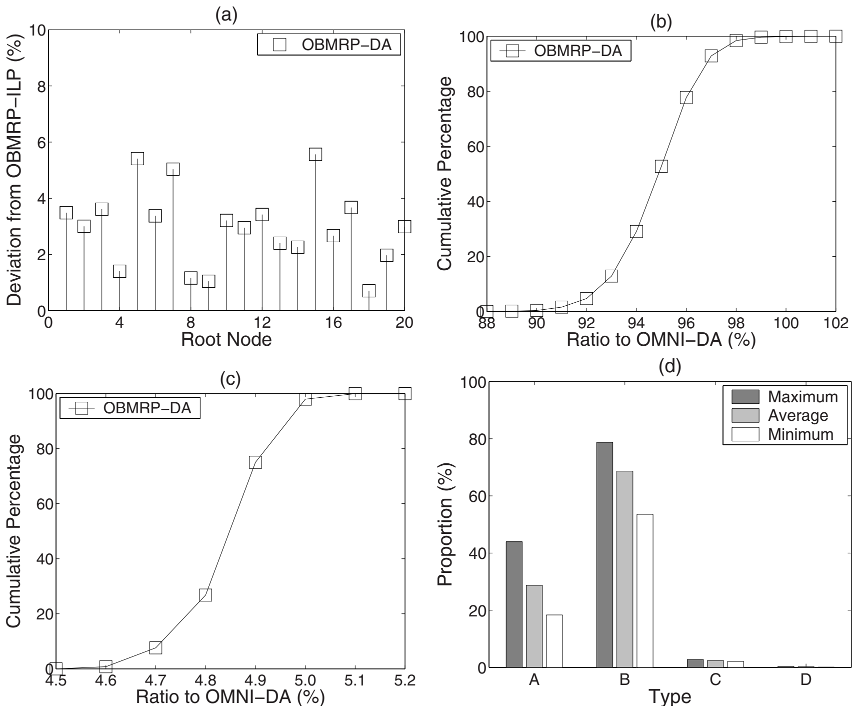

Fig. 3. Efficacy of OBMRP-DA: (a) $\bar{L}$ performance compared with OBMRP-ILP; (b) $\bar{L}$ performance compared with OMNI-DA; (c) Control overhead compared with OMNI-DA; (d) Distribution of transformation operations

These results clearly demonstrate the efficacy of OBMRP-DA. In general, OBMRP-DA converges to better quality solutions than OMNI-DA. In only four out of the 3,000 simulation experiments, the $\bar{L}$ performance of OBMRP-DA is slightly worse than OMNI-DA. On average, OBMRP-DA outperforms OMNI-DA by more than 5%. However, as shown in Fig. 3(c), OBMRP-DA requires on average only 5% of the control overhead as required by OMNI-DA. This is due to the fact that OMNI-DA relies on a set of five transformation operations which in general require probing among all nodes within up to two levels of each other. In contrast, even the most elaborate Type-D transformation defined in OBMRP-DA merely requires probing between node i and all nodes within one level of node j for a particular tree restructuring request $\overrightarrow{i,j}$. Moreover, this is also due to the fact that, for each particular tree restructuring request, OMNI-DA attempts all five transformation operations until one of them is successful. In contrast, OBMRP-DA performs no more than one transformation, and requires only a modicum of the elaborate Type-D transformation operation, as shown in Fig. 3(d), to achieve the comparable $\bar{L}$ performance.

5 Conclusion and Future Work

In this paper, we have identified and addressed a strongly motivated overlay backbone multicast routing problem to support large-scale live Internet broadcasting services in CDNs more efficiently using the two-tier overlay multicast architecture. We have proposed a lightweight algorithm for solving OBMRP in a distributed iterative way. Simulation experiments have confirmed that our proposed algorithm can yield good quality solutions with small control overhead. Our future work is to design a distributed protocol based on the lightweight algorithm proposed in this paper, and to test the multicast routing performance in real or emulated CDN environments.

Acknowledgments. This project is supported by Australian Research Council (ARC) Discovery Grant DP0557519.

References

1. Sripanidkulchai, K., Maggs, B., Zhang, H.: An analysis of live streaming workloads on the Internet. In: Proc. ACM IMC 2004, October 2004, pp. 41–54 (2004)
2. Dilley, J., Maggs, B., Parikh, J., Prokop, H., Sitaraman, R., Weihl, B.: Globally distributed content delivery. IEEE Internet Comput. 6(5), 50–58 (2002)
3. Deering, S.E., Cheriton, D.R.: Multicast routing in datagram internetworks and extended LANs. ACM Trans. Comput. Syst. 8(2), 85–110 (1990)
4. Diot, C., Levine, B.N., Lyles, B., Kassem, H., Balensiefen, D.: Deployment issues for the IP multicast service and architecture. IEEE Network 14(1), 78–88 (2000)
5. Chu, Y.H., Rao, S.G., Zhang, H.: A case for end system multicast. In: Proc. ACM SIGMETRICS 2000, Santa Clara, CA, USA, June 2000, pp. 1–12 (2000)
6. Kwon, M., Fahmy, S.: Topology-aware overlay networks for group communication. In: Proc. ACM NOSSDAV 2002, Miami, FL, USA, May 2002, pp. 127–136 (2002)
7. Zhang, B., Jamin, S., Zhang, L.: Host multicast: A framework for delivering multicast to end users. In: Proc. IEEE INFOCOM 2002, New York, USA, June 2002, vol. 3, pp. 1366–1375 (2002)
8. Banerjee, S., Bhattacharjee, B., Kommareddy, C.: Scalable application layer multicast. In: Proc. ACM SIGCOMM 2002, Pittsburgh, PA, USA, August 2002, pp. 205–217 (2002)
9. Malouch, N.M., Liu, Z., Rubenstein, D., Sahu, S.: A graph theoretic approach to bounding delay in proxy-assisted, end-system multicast. In: Proc. IEEE IWQoS 2002, May 2002, pp. 106–115 (2002)
10. Shi, S.Y., Turner, J.S.: Routing in overlay multicast networks. In: Proc. IEEE INFOCOM 2002, New York, USA, June 2002, vol. 3, pp. 1200–1208 (2002)
11. 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 2003, San Francisco, CA, USA, March 2003, pp. 1521–1531 (2003)
12. Lao, L., Cui, J.H., Gerla, M.: TOMA: A viable solution for large-scale multicast service support. In: Boutaba, R., Almeroth, K.C., Puigjaner, R., Shen, S., Black, J.P. (eds.) NETWORKING 2005. LNCS, vol. 3462, pp. 906–917. Springer, Heidelberg (2005)

13. Guo, J., Jha, S.: Host-aware routing in multicast overlay backbone. In: Proc. IEEE/IFIP NOMS 2008 (2008)
14. Hei, X., Liang, C., Liang, J., Liu, Y., Ross, K.W.: Insights into PPLive: A measurement study of a large-scale P2P IPTV system. In: Proc. Workshop on Internet Protocol TV (IPTV) services over World Wide Web in conjunction with WWW 2006, Edinburgh, Scotland (May 2006)
15. Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness. W. H. Freeman, San Francisco (1979)
16. Zegura, E.W., Calvert, K.L., Bhattacharjee, S.: How to model an internetwork. In: Proc. IEEE INFOCOM 1996, San Francisco, CA, USA, March 1996, vol. 2, pp. 594–602 (1996)

Appendix

Here we provide an ILP formulation for OBMRP. Let the 0-1 variables $p_{i,j}^{v}$, $v \in V - \{r\}$, $\langle i,j \rangle \in E$, indicate if the directed edge $\langle i,j \rangle$ is included in the overlay routing path from node r to node v. Let the 0-1 variables $t_{i,j}$, $\langle i,j \rangle \in E$, indicate if the directed edge $\langle i,j \rangle$ is included in the overlay backbone multicast tree. OBMRP can be mathematically formulated as:

$$\text{Minimize} \quad \sum_{v \in V-\{r\}} w_v \sum_{\langle i,j \rangle \in E} p_{i,j}^{v} l_{i,j} \tag{17}$$

subject to

$$\sum_{j:\langle i,j \rangle \in E} p_{i,j}^{v} - \sum_{j:\langle j,i \rangle \in E} p_{j,i}^{v} = \begin{cases} 1, \text{ if } i = r \\ 0, \text{ if } i \in V - \{r, v\} \\ -1, \text{ if } i = v \end{cases}, \ \forall v \in V - \{r\} \tag{18}$$

$$\sum_{v \in V-\{r\}} p_{i,j}^{v} \le (N-1) \cdot t_{i,j}, \ \forall \langle i,j \rangle \in E \tag{19}$$

$$\sum_{\langle i,j \rangle \in E} t_{i,j} = N - 1 \tag{20}$$

$$\sum_{j:\langle i,j \rangle \in E} t_{i,j} \le \widetilde{d}_i, \ \forall i \in V \tag{21}$$

$$\sum_{\langle i,j \rangle \in E} p_{i,j}^{v} l_{i,j} + \gamma_v \le L_{\max}^{\mathrm{B}}, \ \forall v \in V - \{r\} \tag{22}$$

The objective function in (17) is equivalent to (3) by the definition of $p_{i,j}^{v}$. Equations (18) and (19) ensure that the solution is a directed spanning tree rooted at node r. More explicitly, they enforce one single overlay routing path for each source-sink pair. Equation (20) restricts that the sum of out-degrees counts $N-1$. Equation (21) enforces the residual degree constraint. Equation (22) ensures that the de facto maximal end-to-end latency of the overlay backbone multicast tree is bounded by $L_{\max}^{\mathrm{B}}$. All equations jointly ensure that the solution is a directed spanning tree rooted at node r and satisfies all constraints of OBMRP.