Overlay Multicast with Inferred Link Capacity Correlations

Ying Zhu, Baochun Li
Department of Electrical and Computer Engineering
University of Toronto
{*yz, bli*}*@eecg.toronto.edu*

Abstract

The mapping of overlay links to paths in the underlying network causes the phenomenon of multiple overlay links sharing bottleneck physical links, which leads to correlation of overlay link capacities. We integrate these link capacity correlations by modeling the overlay using linear capacity constraints — which accurately and succinctly capture the link correlation information. It is demonstrated through analysis and simulations that this model is more accurate in representing the true network topology and hence permits higher-bandwidth multicast.

We identify the problem of overlay multicast in the context of our model of overlays with linear capacity constraints (LCC). We show that finding a multicast tree in an overlay network with LCC is NP-complete. We design an efficient heuristics algorithm to solve the problem. Extensive simulations show that our algorithm is able to construct multicast trees that are optimal or extremely close to optimal, with significantly higher bandwidth than trees formed in overlays with no LCC. We further show that multicast trees obtained by our algorithm with a restricted and inherently distributed class of LCC — node-based LCC — also exhibit near-optimal performance.

Index Terms

Overlay networks, algorithms, network protocols.

Technical area: Network Protocols

I. Introduction

Since it was first proposed by Chu *et al.* [1], application-layer overlay multicast has been widely studied, e.g., [2], [3], and has established itself as a truly viable and pragmatic alternative to IP multicast. Overlay networks are distinct from regular flat networks, such as the IP network, in having a two-level hierarchical structure. The high-level is an overlay consisting of end systems as nodes and unicast connections as links; the low-level is a densely connected IP network of routers and physical links. Overlay links map to paths in the low-level network. When overlay links map to paths that share a common physical link, the *sum* of the capacities of the overlay links is constrained by the capacity of the shared physical link. We say these overlay links are *correlated* in capacity.

Some existing work, however, views the overlay as a regular network graph where capacities of links are simply scalars which are their unicast bandwidths. We refer to this as the independent overlay model, as it models overlay link capacities as uncorrelated, independent entities. In these work, although a node degree bound is typically imposed to alleviate overloading of low-level links, there is no explicit incorporation of link correlations. Other related work considers *location-aware* or *topology-aware* overlays (*e.g.*, [4], [5]). They either focus on end-to-end latencies, or assume that the underlying IP-layer topology is known. In comparison, we seek to focus on end-to-end bandwidth, and believe that it is prohibitively expensive — and thus not feasible — to discover the entire AS-level topology using overlay probing techniques (such as `traceroute`), just for the sake of improving overlay performance.

We introduce *linear capacity constraints* (LCC) as a formulation of link correlations in overlays. Instead of scalars, the overlay links are assigned capacity variables and linear constraints of these variables express the hidden link correlations. For instance, suppose two overlay links u and v map to paths that share the physical link e with capacity $c(e)$, this link correlation would give rise to the linear capacity constraint $x_u + x_v \leq c(e)$, where x_u, x_v are the respective capacity variables for u, v. The LCC-overlay is simply an overlay graph with variables for the link capacities together with a set of LCC.

In this paper, we study the problem of optimizing bandwidth in constructing overlay multicast trees. We define three metrics to measure the performance and quality of an overlay multicast tree. These metrics evaluate how accurately the overlay tree characterizes the underlying network topology, how close to the optimal is the tree bandwidth, and how much the tree overloads the low-level links. We show that the optimal multicast tree found in an independent overlay cannot reliably achieve high bandwidth, due to its fundamental weakness of being unable to accurately represent the true network topology.

We formulate the new problem of overlay multicast in the LCC-overlay: Given an overlay with a set of LCC, how do we find a highest-bandwidth multicast tree? We show that this problem is NP-complete. We propose a heuristics algorithm to solve this problem.

Two types of LCC are considered: complete LCC and node-based LCC. Complete LCC is the set of LCC that completely captures all the link correlations. Since this requires global information, we also study node-based LCC, in which each constraint contains only those links that are incident to the same node. Through extensive simulations, we compare the performance of multicast trees constructed in the independent overlay, and those found by our algorithm using complete LCC and using node-based LCC. Independent overlay multicast performs relatively

poorly. Our algorithm always obtains near-optimal trees, not only with complete LCC, but even with the restricted and inherently distributed node-based LCC. Simulation results show that the algorithm converges quickly and is scalable for increasing network sizes. Furthermore, we discuss distributed construction of an LCC-overlay with node-based LCC by employing certain probing techniques, and propose a distributed variation of our algorithm.

Remark. We observe that to ensure high-bandwidth overlay multicast, the overlay network must have awareness and knowledge of the underlying network topology. Rather than attempting to acquire complete information of the underlying topology, the formulation of LCC to express overlay link correlations is an *efficient and highly condensed* mechanism for characterizing topological capacity information. Certainly, LCC do not contain complete topological information of the underlying network; however, LCC provide *sufficient* information to capture overlay link correlations in capacity, thereby enabling overlay multicast that has bandwidth negligibly close to the optimal. A surprising and encouraging result from our investigation of LCC is that even with the incomplete node-based LCC, near-optimal bandwidth multicast is ensured. It should be remarked that node-based LCC can be obtained by limited simple probing and every overlay node need only probe its neighboring (or incident) links. It is only owing to LCC that with such relatively small overhead and complexity, one can always achieve near-optimal overlay multicast.

The remainder of the paper is organized as follows. We summarize related work and distinguish our work in Sec. II. In Sec. III, we introduce the notion of LCC and present the formal definitions of LCC and the metrics. The problem of finding high-bandwidth multicast trees is studied in Sec. IV, as well as the description of the heuristics algorithm. We evaluate the algorithm in Sec. V, and discuss distributed construction of LCC-overlays and multicast trees in Sec. VI. The paper is concluded in Sec. VII.

II. Related Work

We have shown that through accurate network representation, LCC-overlay ensures much better bandwidth in data dissemination. To the best of our knowledge, there has not been previous work on the problem of highest-bandwidth multicast tree with LCC that we studied in this paper.

Structured overlay networks are constructed based on Distributed Hash Tables (*e.g.,* Chord [6]). Distributed algorithms for general-purpose overlay construction were proposed by Young *et al.* in [7] and by Shen in [8]. Application-specific proposals have been made for various overlay services, including overlay multicast [1], [2], content distribution [9], [10] and multimedia streaming [11], [3]. Also relevant is previous work by Ratnasamy *et*

al. [4]. A distributed binning scheme is designed and applied to the formation of unstructured overlay networks. The aim is to incorporate more topological awareness into overlay construction. This work differs from ours in focusing exclusively on the latency metric.

Common to all these proposals are heuristics that use unicast probing to select overlay routes with low latency or high bandwidth. They view and treat overlay links as independent. However, we have studied the new problem of high-bandwidth data dissemination in a multicast tree within the framework of overlays with linear capacity constraints. In our previous paper [12], we only studied problem of overlay unicast (widest path and maximum flow). As well, in this paper, the metrics are defined differently with respect to multicast tree, and the stress metric is new.

Recent work by Kim *et al.* [13] proposes a protocol to eliminate probed bottlenecks in an overlay multicast tree. Our work is distinct from theirs in formalizing link correlations using linear capacity constraints and thus being able to find an efficient and distributed algorithm based on node-based LCC. We also rigorously define the metrics of accuracy, efficiency and stress, which not only quantitatively measure performance, but also provide crucial insight into the quality of overlay networks in general. By formalizing the LCC-overlay model, and by conceptually separating LCC-overlay construction and finding multicast trees using LCC, we are able to (1) prove that finding the optimal overlay multicast tree is NP-complete, and (2) develop fully distributed algorithms which consistently achieve performance that is remarkably close to the optimal. Our evaluation of the performance is based on comparing against a rigorously determined upper bound of the optimal.

III. Linear Capacity Constraints in Overlay Networks

We will first introduce the concept of linear capacity constraints (LCC) in overlay networks and demonstrate the advantage of the LCC overlay model in improving overlay quality and thereby bandwidth. Then we will formally define the LCC-overlay and several metrics that measure the quality of overlay multicast trees.

By their nature, overlay networks differ from the usual IP network. An overlay is a hierarchical network comprising two levels: the lower level is the IP network of routers and physical links, while the higher level is an application-layer network of end-systems (overlay nodes) and virtual links (overlay links). A virtual link is the unicast connection between two overlay nodes; it maps to a path of physical links in the low-level network, which is determined by IP routing.

It is possible that two overlay links map to paths which share a bottleneck physical link. Obviously, the sum of the capacities of these overlay links is bounded by the capacity of the underlying bottleneck link. We say these two overlay links are *correlated* in capacity. Our proposal is to model the overlay by explicitly expressing link correlations as linear capacity constraints (LCC). We will show that the *LCC overlay model* is necessary for ensuring overlay quality, in particular, for yielding the highest bandwidth in overlay data dissemination.

Previous proposals of algorithms and protocols for overlay networks have adopted, implicitly or explicitly, a model of the overlay as a traditional flat network rather than hierarchical. That is, they do not specifically take into consideration overlay link correlations; they assume that overlay links are all independent. Henceforth we will refer to this as the *independent overlay* model.

In the independent model, an overlay network is viewed as a weighted graph where the edges are overlay links weighted by their unicast bandwidth and delay. A generic scheme of constructing an overlay with the aim of optimizing bandwidth is: each node selects its neighbors by choosing k adjacent links with the highest unicast bandwidth. The imposition of a node degree limit k is a commonly used heuristic to alleviate overloading in the low-level network. Since this is representative of overlay construction schemes from previous work, we use it below to assess the independent model.

We use the example in Figure 1 to illustrate the disadvantage of the independent model, and to introduce and show the advantage of the LCC model.

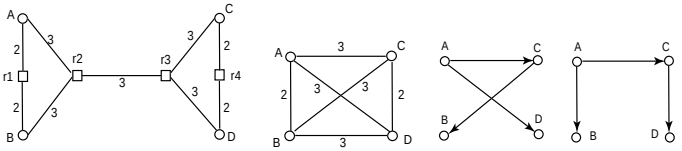


Fig. 1. A simple example illustrating the disadvantage of the independent model and the advantage of linear capacity constraints.

The example network in Figure 1(a) contains four overlay nodes, A, B, C, D, and four low-level routers, $r1, r2, r3, r4$. The edges shown are physical links and labeled by their bandwidths. The paths to which overlay links correspond are the obvious ones (e.g., $(A, B) = \langle A, r1, B\rangle; (A, C) = \langle A, r2, r3, C\rangle; (B, C) = \langle B, r2, r3, C\rangle$).

The overlay graph is given in Figure 1(b), in which the overlay links are labeled by their independent unicast bandwidths. Figure 1(b) is the case when k is 3 in the above construction scheme based on the independent model; it is not hard to see that the following reasoning will also hold for $k = 2$ or 1.

In the independent model, the highest-bandwidth multicast tree in this overlay graph[1] is shown in Figure 1(c).

[1]The tree can be found by an all-widest paths algorithm that is a variant of Dijkstra's all-shortest paths.

Although the predicted bandwidth of the tree is 3, it can be seen that the achievable bandwidth of the tree is 1 because all three links in the tree share a single bottleneck physical link $(r2, r3)$ with capacity 3.

In the LCC model, instead of links labeled by numbers, the capacity of each overlay link is represented by a variable and a set of linear constraints are used to formulate link correlations. For instance, the above link correlation among the three overlay links can be easily captured by the constraint $x_{AC} + x_{AD} + x_{CB} \leq 3$, where x_{AC} is the capacity variable for link (A, C) and so on. The complete set of linear capacity constraints (LCC) is given below in matrix form:

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_{AB} \\ x_{AC} \\ x_{AD} \\ x_{BC} \\ x_{BD} \\ x_{CD} \end{pmatrix} \leq \begin{pmatrix} 2 \\ 3 \\ 2 \end{pmatrix} \tag{1}$$

The LCC-overlay is the overlay graph with link capacity variables together with the set of LCC. To find a highest-bandwidth multicast tree in this LCC-overlay, one can devise a simple variant of the algorithm for regular graphs. The difference is that every time a link is considered for selection, its capacity (instead of a fixed number) is computed as its fair share of the capacities in the LCC with the set of already selected links. The tree thus obtained is given in Figure 1(d). In this case, the achievable bandwidth is 2, the same as the predicted bandwidth.

The highest-bandwidth tree obtained in the LCC-overlay has twice the bandwidth of that in the independent overlay. Moreover, the LCC tree in fact achieves the optimum multicast tree bandwidth, by the following reasoning. At least one of the four links, $(A, C), (A, D), (B, C), (B, D)$, must be included in any multicast tree to ensure connectivity; yet, if more than one of them are chosen, then the tree bandwidth would be at most $3/2$. Thus an optimum tree contains exactly one of these four links, which implies that both links (A, B) and (C, D) must be in the tree in order to span all overlay nodes —the optimum bandwidth is 2.

From the above example and analysis, we can observe that the achievable bandwidth of the independent overlay is low because it does not explicitly incorporate link correlations and therefore has insufficient information to obtain a high-bandwidth tree. The achievable bandwidth being only $1/3$ of the predicted bandwidth is ample evidence that the independent overlay (1) lacks accuracy in representing the true network topology and (2) as a result, cannot attain high bandwidth. In the LCC-overlay, however, the incorporation of link correlations formulated as LCC

results in an accurate representation of underlying topology, and as a consequence, an optimal multicast tree can be found.

A. Formal definitions of LCC and metrics

We now present formal definitions of LCC-overlay and several metrics that measure the performance and quality of overlay multicast trees.

The two-level hierarchy of an overlay network can be formulated as consisting of:

- A low-level (IP) graph $G = (V, E)$; each low-level link $e \in E$ has a capacity of $c(e) \geq 0$.
- A high-level overlay graph $\widehat{G} = (\widehat{V}, \widehat{E})$, where $\widehat{V} \subset V$;
- A mapping P of every overlay edge $(\widehat{v}_1, \widehat{v}_2) \in \widehat{E}$ to a low-level path $P(\widehat{v}_1, \widehat{v}_2) \subset G$ from $\widehat{v}_1$ to $\widehat{v}_2$.

The formulation of capacity constraints in the overlay graph $\widehat{G}$ is where LCC-overlay departs from independent overlay. The independent overlay is defined as follows.

Definition 1 (Independent overlay): The *independent overlay* is a pair $(\widehat{G}, \widehat{c})$, where $\widehat{c}$ is a capacity function such that each link $\widehat{e} \in \widehat{E}$ has a nonnegative capacity $\widehat{c}(\widehat{e}) \geq 0$.

The LCC-overlay is defined as follows.

Definition 2 (LCC-overlay): The *LCC-overlay* is a triplet $(\widehat{G}, C, b)$ where

- The capacity of each link $\widehat{e}$ in $\widehat{G}$ is a variable $x_{\widehat{e}}$.
- (C, b) represent a set of m linear capacity constraints $Cx \leq b$:
  - C is a 0-1 coefficient matrix of size $m \times |\widehat{E}|$;
  - x is the $|\widehat{E}| \times 1$ vector of link capacity variables;
  - $b \in \mathrm{R}^m$ is the capacity vector.

Each row i in (C, b) is a constraint of the form $\sum_{\widehat{e}:C(i,\widehat{e})=1} x_{\widehat{e}} \leq b(i)$.

1) Overlay (or predicted) bandwidth: Given any tree T in either the independent overlay or the LCC-overlay, we observe that there exists the notion of the *overlay bandwidth* of T. The overlay bandwidth of a tree can be thought of as its *predicted bandwidth* in a certain overlay model. Because tree construction algorithms in overlays rely solely on predicted bandwidth to optimize tree formation, this is an crucial concept to clarify and study in these two distinct overlay models.

For a tree T in the independent overlay $(\widehat{G}, \widehat{c})$, the overlay bandwidth of T is the minimum bandwidth link in T as given by $\widehat{c}$, or formally, $\min\{\widehat{c}(\widehat{e}) : \widehat{e} \in T\}$.

As for a tree T in an LCC-overlay $(\widehat{G}, C, b)$, the overlay bandwidth of T can be obtained by the following procedure. For each constraint (C_i, b_i) (i-th row in C, b), compute the bandwidth allocated to every link $j \in T$ for which $C_{i,j} = 1$: $\gamma_i(j) = b_i / \sum_{k \in T} C_{i,k}$. That is, $\gamma_i(j)$ is the bandwidth allocated to j in T by the i-th constraint. Considering all constraints together, the allocated bandwidth of link j in T is $\gamma j = \min\{\gamma_i(j) : i = 1 \ldots m\}$, where m is number of rows in C. Finally, the overlay bandwidth of T is $\sigma(T) = \min\{\gamma(j) : j \in T\}$.

2) Achievable bandwidth $\sigma_G(T)$: Another notion of interest is: Given any overlay tree T spanning all overlay nodes, what is the *achievable bandwidth* of T? Let $\sigma_G(T)$ denote the achievable bandwidth of T in overlay $\widehat{G}$ residing on top of G. First, for each low-level edge e, equal shares of its capacity are allocated to overlay edges in T. It follows that an overlay edge $\widehat{e}$ has a bandwidth allocation from every low-level edge to which $\widehat{e}$ maps. The achievable bandwidth of $\widehat{e}$ is the minimum of all these allocations. Finally, the minimum of achievable bandwidths of all overlay edges in T is the achievable bandwidth of T, $\sigma_G(T)$. The procedure for obtaining $\sigma_G(T)$ is summarized in Figure 2. The following verifies the correctness of the procedure.

Proposition: Given an overlay $\langle \widehat{G}, G, P \rangle$ and any overlay tree $T \subseteq \widehat{G}$, the highest bandwidth that T can achieve is $\sigma_G(T)$ obtained by the procedure in Fig. 2.

Proof: For each overlay edge $\widehat{e}$ in T, its bandwidth $\gamma_T(\widehat{e})$ is assigned the minimum of its allocations from low-level edges mapped from $\widehat{e}$. Consequently, at each low-level edge e, the γ_T of each mapped overlay edge is no greater than its allocation, hence the sum of γ_T's cannot exceed the capacity of e. Since $\sigma_G(T)$ is the minimum of the γ_T's, it is obvious that $\sigma_G(T)$ does not violate any low-level capacity and is hence feasible.

Because the bandwidth metric is concave and the bandwidth of a tree is the minimum bandwidth of its edges, the achievable bandwidth of T must be no greater than the minimum of all allocations given out by low-level edges. To maximize bandwidth of T, one is actually maximizing the minimum of the allocations by low-level edges. It is easy to see that to maximize the minimum, a low-level edge should assign equal allocations. Therefore, the $\sigma_G(T)$ obtained by the procedure is the highest bandwidth T can achieve. □

```
for each e ∈ E
   Allocate c(e) equally among
     {ê : e ∈ P(ê) and T(ê) > 0},
   let each allocation be denoted by γ^e_T(ê)
   for each ê ∈ Ê
     if T(ê) > 0    γ_T(ê) ← min{γ^e_T(ê) : e ∈ P(ê)}
     else           γ_T(ê) ← 0
   σ_G(T) ← min{γ_T(ê) : γ_T(ê) ≠ 0 and ê ∈ Ê}
```

Fig. 2. The procedure for obtaining achievable bandwidth of a tree T in G, $\sigma_G(T)$.

We proceed now to present definitions of three metrics: accuracy, efficiency and stress. Accuracy is the ratio of the overlay or predicted bandwidth of a tree over its achievable bandwidth. Efficiency is the ratio of achievable

bandwidth of a tree over the optimum tree bandwidth. Stress is defined for low-level links that are mapped by overlay tree links; it is essentially the load placed on a low-level link by the overlay tree, normalized by the link capacity. The formal definitions are given below.

Definition 3 (Accuracy): The *accuracy* of a multicast tree T in an overlay network $\langle G, \widehat{G}, P\rangle$, is

$$\frac{\text{overlay bandwidth of } T \text{ in } \widehat{G}}{\sigma_G(T)}. \tag{2}$$

Definition 4 (Efficiency): The *efficiency* of a multicast tree T in an overlay network $\langle G, \widehat{G}, P\rangle$, is

$$\frac{\sigma_G(T)}{\sigma_G(T_{\text{opt}})}, \tag{3}$$

where T_{opt} is an optimum multicast tree in $\langle G, \widehat{G}, P\rangle$.

Definition 5 (Stress): The *stress* of a low-level edge $e \in E$ due to a multicast tree T in an overlay network $\langle G, \widehat{G}, P\rangle$, is

$$\frac{|\{\widehat{e} : \widehat{e} \in \widehat{E} \text{ and } \widehat{e} \in T\}|}{c(e)}, \tag{4}$$

where $c(e)$ is the capacity of e.

Remark: It should be clarified here that our definition of *stress* is original. In [1], the stress metric was defined with respect to a physical link: it is the number of overlay links mapped to the physical link. The problem with this definition is that it does not take into consideration at all the capacity of the physical link. It makes no distinction between physical links with large capacity and those with small capacity. Obviously, even with more overlay links mapped to it, a physical link with a huge capacity can still provide higher bandwidth to each overlay link, compared to a physical link with small capacity that has only a few overlay links mapped to it. According to the definition of stress in the previous work [1], the 'stress' of the small-capacity link would be lower than the 'stress' of the large-capacity link, and therefore, the former should be preferred when building the multicast tree. However, since the goal is to maximize bandwidth, the large-capacity link should be preferred despite its higher 'stress' value. By this reasoning, we incorporate capacity into our stress definition, by dividing the number of overlay links by the capacity of the physical link.

IV. Multicast Tree with Linear Capacity Constraints

A. MTC is NP-complete

We showed above that by using LCC, the bandwidth of multicast trees can be significantly increased. Now we are faced with the question of how to obtain the highest-bandwidth multicast tree in an LCC-graph.

We state the problem of highest-bandwidth multicast tree with LCC (MTC) as a decision problem as follows.

INSTANCE: An LCC-graph (G, C, b), where $G = (V, E)$ and (C, b) are LCC; a positive integer B.

QUESTION: Is there a multicast tree T for G such that the bandwidth of T is $\geq B$?

Theorem: MTC is NP-complete.

Proof:

MTC is in NP because given an LCC-graph (G, C, b), a positive integer B and a multicast tree T for G, the bandwidth T can be computed by the polynomial-time procedure in Sec. III-A for obtaining the overlay bandwidth of T in an LCC-overlay.

To show that MTC is NP-complete, we reduce the NP-complete problem Degree-Constrained Spanning Tree (DST) [14] to MTC; the problem DST is defined as:

INSTANCE: Graph $G = (V, E)$, positive integer $K \leq |V|$.

QUESTION: Is there a spanning tree for G in which no vertex has degree larger than K?

Given an instance of DST, $G = (V, E)$ and K, we transform it to an instance of MTC. We let G remain the same and add some LCC. For every node $u \in V$, a constraint is formed: $\sum_{e \in \text{inc}(u)} x_e \leq K \cdot B$, where $\text{inc}(u)$ denotes the set of edges incident to u. This transformation can obviously be done in polynomial time.

Suppose a spanning tree T exists in G for which no node has degree larger than K. That is, $|\text{inc}(u)| \leq K, \forall u \in V$. Then assigning B to each edge in T implies that $x_e = B$ or 0. It follows that $\sum_{e \in \text{inc}(u)} x_e \leq K \cdot B, \forall u \in V$. Thus T is a multicast tree that satisfies all the LCC and $\sigma(T) = B$.

In the other direction, suppose a multicast tree T has bandwidth $B' \geq B$. If a node $u \in T$ has degree greater than K, *i.e.*, $|\text{inc}(u)| = d > K$, then $\sum_{e \in \text{inc}(u)} x_e \geq d \cdot B' > K \cdot B$, which violates the constraint imposed. Therefore no node has degree greater than K. Clearly, T is a solution of DST. □

B. Heuristics Algorithm

In Sec. IV-A above, we showed that the problem of finding the highest-bandwidth multicast tree with LCC is NP-complete. Therefore, we propose a heuristics algorithm to solve the problem in this section. The evaluation of

the algorithm will be deferred to Sec. V.

```
HMTC(Ĝ, C, b)
 1  obtain G with independent edge
     capacities from Ĝ and LCC (C, b)
 2  T_0 ← highest-bandwidth tree in G
 3  (C_{T_0}, b) ← LCC of T_0
 4  R ← edges in T_0 in ascending order
     of bandwidths allocated by (C_{T_0}, b)
 5  T_hi ← T_0
 6  iter ← 1
 7  while iter ≤ max_num_iterations and R ≠ ∅
 8    r ← 1st edge in R
 9    R ← R − {r}
10    T_sub ← subtree under r
11    T ← T_hi − {r}
12    for each edge e ∈ T − T_sub
13      T' ← T ∪ {e}
14      C_{T'} ← LCC of T'
15      if bandwidth(T') > bandwidth(T_hi)
16        T_hi ← T'
17        R ← edges in T_hi in ascending
18          order of allocated bandwidths
19        break out of for-loop
20    iter ← iter + 1
21  return T_hi
```

Fig. 3. Summary of heuristics algorithm *HMTC*.

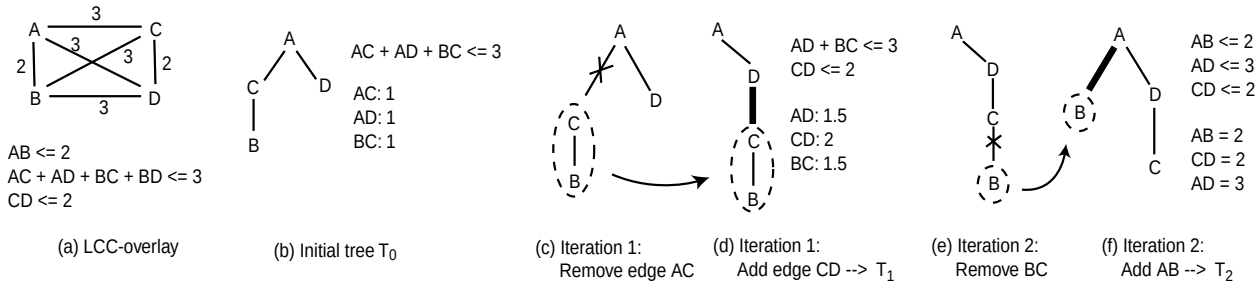


Fig. 4. An example of the heuristics algorithm being executed.

The input is an LCC-graph $(\widehat{G}, C, b)$. The goal is to find a high-bandwidth multicast tree. The main idea of the algorithm is to begin with an initial tree, and make incremental improvements by replacing, at each iteration, the lowest-bandwidth edge with a higher-bandwidth one, thereby increasing bandwidth of the tree. For every edge replacement, the algorithm preserves the topology of a multicast tree, *i.e.*, a tree spanning all receiver nodes.

The initial multicast tree is found by first forming a regular graph G with edges weighted with independent edge capacities (numbers). This is easily done by setting $\min\{b_i : C(i, e) = 1\}$ to be the capacity for each edge $e \in \widehat{G}$. Note that this corresponds exactly to an independent overlay with unicast bandwidths for the edges. A highest-bandwidth multicast tree is found in G and set to be the initial tree, let it be denoted T_0. The algorithm then incrementally improves the tree by always replacing the worst edge with a better edge, if possible, while maintaining the spanning tree structure.

The algorithm, which we call *HMTC* (High-bandwidth Multicast Tree with LCC), is summarized in Table 3.

We re-use the example overlay network in Figure 1 from Sec. III to illustrate the algorithm. The overlay graph along with the LCC (C, b) are in Figure 4(a). For simplicity of presentation, we let UV denote the capacity variable of edge (U, V).

In the graph in Figure 4(a), the numbers labeling the edges are the independent edge capacities. For the graph with independent edge capacities, a highest-bandwidth multicast tree is found: T_0 in Figure 4(b). The *LCC of* T_0 is listed on the right of T_0 in the figure. The LCC of T_0, C_{T_0}, is LCC (of the graph) (C, b) intersected with T_0 so that they only contain capacity variables of edges from T_0. In this instance, there is only one constraint in LCC of T_0: $AC + AD + BC \leq 3$. Each T_0 edge is thus allocated an equal bandwidth of 1. The bandwidth of T_0 is the minimum of allocated bandwidths of its edges, in this case, 1.

The worst edge, one with the lowest bandwidth allocated by the LCC of T_0, is selected to be replaced. Since all three edges have lowest bandwidth, any one can be selected, say edge AC. We want to find another edge XY such that by removing AC and adding XY to form T_1, T_1 is a multicast tree and the bandwidth of T_1 (allocated by the LCC of T_1) is higher than the bandwidth of T_0. In other words, XY connects the subtree under AC, *i.e.*, subtree rooted at node C, with the rest of the tree. The resulting new tree also has improved bandwidth.

Edge CD satisfies the above conditions; the removal of AC and the new tree T_1 formed by adding CD can be seen in Figure 4(c) and (d), respectively. The LCC of T_1 and the bandwidths allocated to the edges are given as well. The bandwidth of T_1 is 1.5, an improvement on T_0.

Now the above edge replacement procedure is repeated for T_1. The new tree T_2 is shown in Figure 4(f); its bandwidth is 2. At this point, none of the three edges can be replaced by another edge to improve the tree bandwidth and the algorithm terminates.

C. Effectiveness of HMTC algorithm in minimizing stress

It is not hard to see that with our definition of stress (Eq. 4), minimizing stress maximizes bandwidth. In a multicast tree, the overlay link with the lowest bandwidth determines the bandwidth of the tree, i.e., receiver bandwidth. Therefore, to maximize multicast bandwidth, one should minimize the maximum stress in the underlying network.

The *HMTC* algorithm attempts to accomplish exactly this: minimizing maximum link stress of underlying physical network. To analyze the ability of *HMTC* to minimize stress, we *visually* examine low-level link stress. For ease of visual representation, we find it convenient to slightly modify the definition of stress. Recall that the stress of a

low-level link e is (number of overlay links mapped to e)/(the capacity of e). The modified definition is essentially *stress ratio*: for e, it is the ratio of stress of e over the minimum stress of all low-level links with at least one overlay link mapped to them, rounded to the nearest integer not greater than it. Stress ratio measures stress levels, 1 being the lowest; for instance, 3 means the links has three times the stress of a link with stress ratio 1. Henceforth in this section, we will refer to stress ratio as stress, since the former is only a certain normalized form of the latter.

Visual representation of link stress is accomplished by *stress graphs*. A stress graph is a graph of low-level links that have overlay links mapped to them, and for each link between a pair of nodes (u, v), the number of edges connecting u and v in the stress graph is the stress of link (u, v).

Just to illustrate how *HMTC* incrementally minimizes the maximum link stress after each iteration of the algorithm, we revisit our example in Fig. 4 from Sec. IV-B. In Fig. 5, we show the stress graphs of the trees constructed at each iteration of *HMTC*. It can be seen that the worst link stress progressively decreases as the algorithm progresses.

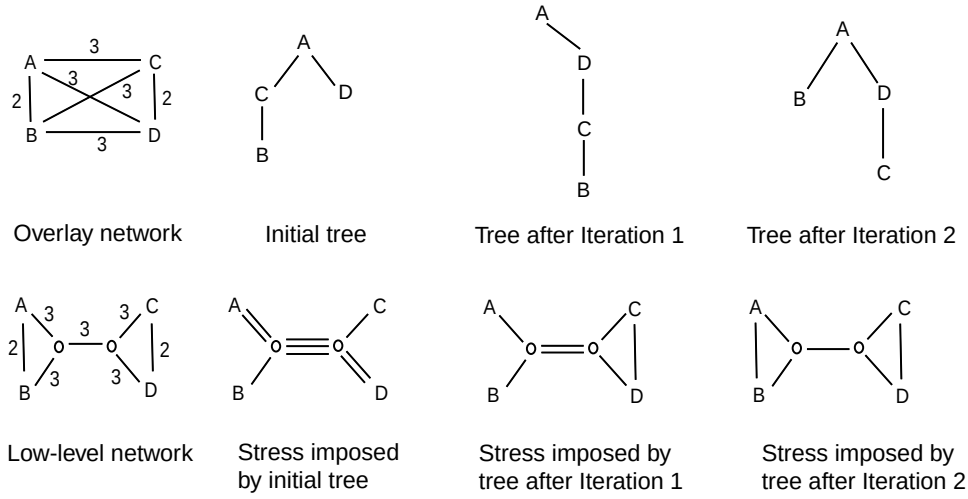
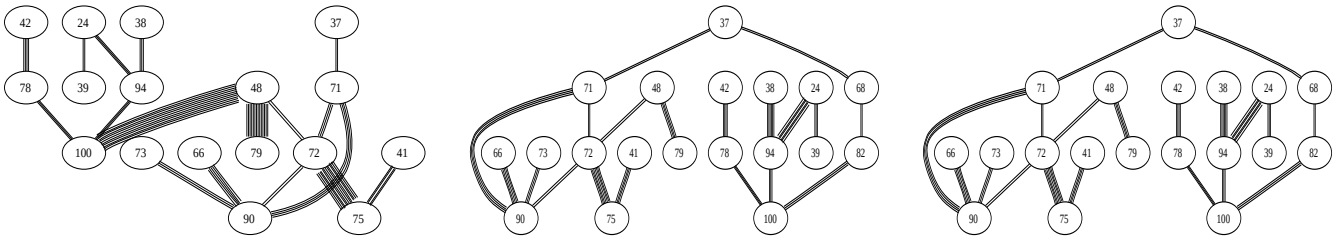


Fig. 5. Incremental improvement in stress throughout execution of *HMTC* on the example network. The top four graphs are overlay and the bottom four are underlying network. (a) Top: example overlay network graph. Bottom: underlying network graph (with two routers in the center). (b) Top: Initial overlay multicast tree. Bottom: Stress graph of low-level links mapped by initial tree. (c) Top: Tree after first iteration. Bottom: Stress graph of low-level links mapped by tree. (c) Top: Tree after second (and final) iteration. Bottom: Stress graph of low-level links mapped by tree.

We conducted simulations on larger, random, realistic networks to see whether *HMTC* indeed does minimize stress consistently. The specific way we generate these topologies is explained in the next section, Sec. V. In order to have a deeper investigation of our algorithm with different types of LCC-overlays, we study both LCC-overlays with a complete set of LCC (*HMTC* with Complete-LCC) and LCC-overlays with a restricted set of node-based LCC (*HMTC* with Node-LCC). Node-based LCC is, as the name suggests, a set of LCC in which each constraint contains only links incident to a single node. For comparison with *HMTC*, we simulate the optimal tree bulit in an independent overlay (i.e., without LCC), without using *HMTC*, referred to as Independent subsequently. For simplicity, we will from now on refer to *HMTC* with Complete-LCC as just Complete-LCC and *HMTC* with

Node-LCC as just Node-LCC.

Here we give one set of representative results (on the smallest-sized network in our simulations) from our simulation results. Sec. V will give a more elaborate and comprehensive presentation of the simulation results.



(a) Stress graph for low-level network mapped to by optimal tree in independent overlay, network size = 100.

(b) Stress graph for low-level network mapped to by multicast tree obtained by *HMTC* with Complete-LCC, network size = 100.

(c) Stress graph for low-level network mapped to by multicast tree obtained by *HMTC* with Node-LCC, network size = 100.

Fig. 6. Stress graphs, network size = 100.

Figure 6 shows the stress graphs of underlying physical networks for an optimal tree in the independent overlay, and trees obtained by *HMTC* with Complete-LCC and *HMTC* with Node-LCC. It is clearly the case that the Independent tree imposes much higher stress, on three physical links in particular, than the two *HMTC* trees obtained using Complete-LCC and Node-LCC. The low-level physical links mapped to by *HMTC* trees have better balanced stress; there are no great variations in the link stress values. This, of course, leads to higher receiver bandwidth, as the efficiency results of the simulations reveal in the next section, Sec. V.

We shall elaborate more on the relationship between minimizing stress and higher efficiency (i.e., multicast bandwidth) in Sec. V, where we evaluate the performance, convergence and scalability of the algorithm.

V. Evaluation of Algorithm Performance and Convergence

In this section, we evaluate the performance and convergence of our heuristics algorithm *HMTC* that finds a high-bandwidth multicast tree in an LCC-overlay. Two types of LCCs are considered: complete LCC —the complete set of LCC that expresses all link correlations, and node-based LCC —every constraint contains only links incident to the same node. We also consider optimal (*i.e.*, highest-bandwidth) multicast trees in independent overlays; for brevity, it is referred to as an optimal independent tree. We compare these three cases: optimal independent tree, *HMTC* with complete-LCC and *HMTC* with node-LCC. To be succinct, we will refer to optimal independent tree as *Independent*, *HMTC* with complete LCC as *Complete-LCC* and *HMTC* with node-based LCC as *Node-LCC*. We will moreover evaluate the convergence and scalability of *HMTC*.

For the purpose of generating realistic network topologies, we use an Internet topology generator, BRITE [15],

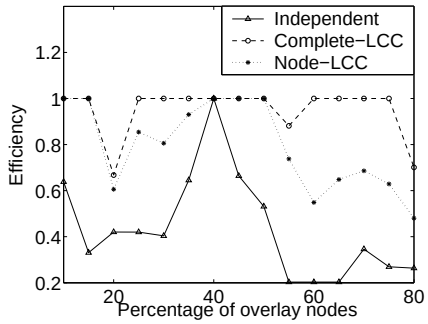


Fig. 7. Efficiency versus overlay ratio, low-level size = 300.

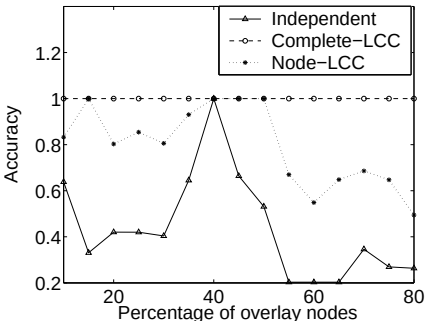


Fig. 8. Accuracy versus overlay ratio, low-level size = 300.

which is based on power-law degree distributions[2]. Networks of various sizes are generated using BRITE and a fraction of the nodes with the lowest degrees (mostly 1 or 2) are selected to be the overlay nodes. Random distribution is used for assigning bandwidth to low-level links.

A. Performance metrics

For measuring performance and quality, the metrics we use are efficiency, accuracy and stress; their definitions are given previously in Sec. III-A. For stress, we continue to use stress ratio (and stress graph) for evaluation purposes, as in Sec. IV-C. Recall that the *efficiency* of a tree T measures how close the bandwidth of T is to the optimal, specifically, it is the ratio of achievable bandwidth of T over optimal multicast tree bandwidth. However, as shown in Sec. IV-A, finding optimal-bandwidth multicast trees in graphs with link capacity correlations —*i.e.*, in overlay networks —is NP-complete. Thus to compute the efficiency metric, instead of the exact optimal which is hard to find, we use an *upper bound* of the optimal: the optimal *independent* tree bandwidth.

To summarize, we evaluate the performance of *HMTC* using the following calculations of the efficiency and accuracy metrics:

- Efficiency of high-level tree T is its achievable bandwidth over the optimal independent tree bandwidth.
- Accuracy of T is its achievable bandwidth over its predicted bandwidth.

[2]A seminal paper [16] revealed that degree distribution in the Internet is a power-law.

- Stress ratio of low-level link e is its stress over the minimum nonzero stress of low-level links.

Again, we will be using the terms 'stress' and 'stress ratio' interchangeably.

B. Efficiency and Accuracy

We compare the efficiency and accuracy of three types of multicast trees: optimal independent trees (Independent), trees given by *HMTC* with complete LCC (Complete-LCC) and with node-based LCC (Node-LCC).

First, we study the effect of different ratios of overlay size to low-level size, by fixing low-level network size to 300 and varying the percentage of overlay nodes from 10% to 80%.

Efficiency for the three types of trees is plotted against the ratio of overlay to low-level size, in Figure 7. The efficiency of Complete-LCC is all 1 except for three instances; it is consistently the highest and always much higher than the Independent efficiency. Obviously, when it is 1, the optimum is achieved. Because an upper bound of the exact optimum (to find which is NP-hard) is the denominator in the efficiency metric, it is unknown whether the optimum is attained in the three instances that are less than 1. It is not surprising that Complete-LCC is the highest, since it possesses perfect knowledge of all link capacity correlations. However, it is worth remarking that the optimal is almost always obtained, this can be attributed to the effectiveness of the heuristics algorithm *HMTC*.

Node-LCC follows Complete-LCC remarkably closely in efficiency for all overlay-to-network ratios of less than 60%. In realistic scenarios, such as in the Internet, overlay nodes are always greatly outnumbered by low-level nodes. Therefore, it is not unreasonable to say that for all realistic overlay percentages, Node-LCC is near-optimal.

Efficiency of Independent trees is by far the lowest, except at 40% overlay where it is 1. It should be noted as well that overlay percentages close to 40% also yield higher efficiency than the rest —in fact, the entire curve resembles roughly a upside-down bell, narrow in the middle and peaking at 40%. The shape of the curve can be explained. When overlay nodes are few compared to the low-level network size, the paths between them are likely long and contain more links, resulting in higher probabilities of the paths intersecting and sharing links. The likelier overlay link correlations are, the higher the probability that some shared low-level link is forced to allocate equally low bandwidth to overlay links mapped to it.

Things are not necessarily rosier with high densities of overlay nodes. Higher percentage of overlay nodes leads to more overlay links mapped to the same low-level network, naturally, collisions increase. Intuitively, there might exist a middle ground where paths are not so long and overlay links are not so dense, such that overloading of bottleneck links is minimized, thereby achieving the optimal. This optimum point seems to be at 40% here.

The plot of accuracy versus overlay percentage can be seen in Figure 8. It is obvious that Complete-LCC is always perfectly accurate (*i.e.*, accuracy of 1), due to its complete link correlation information. An encouraging observation here is that Node-LCC accuracy is greater than 80% for all networks with less than 60% overlay nodes, which is to say, for all realistic overlays. Note that for Independent trees, accuracy is exactly the same as efficiency, with respect to the slightly modified definition of efficiency above in Sec. V-A. The accuracy curves are in general extremely similar to the efficiency, which is evidence that supports our proposition that to ensure high multicast bandwidth, it is *necessary* for an overlay to *accurately* represent the true network topology. The LCC-overlay accomplishes accuracy by way of succinctly expressing overlay link correlations.

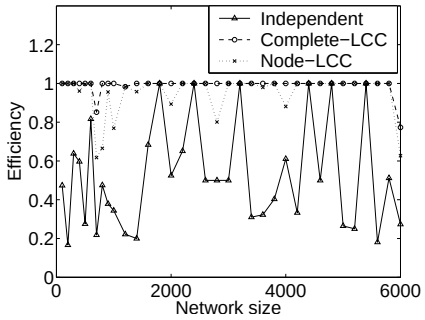


Fig. 9. Efficiency versus network size, 10% overlay nodes.

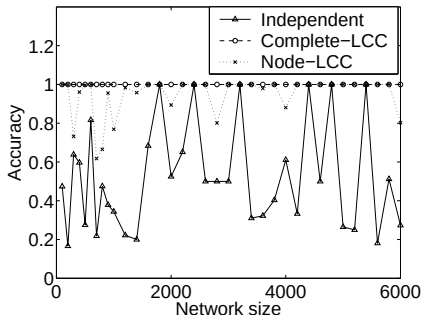


Fig. 10. Accuracy versus network size, 10% overlay nodes.

The next parameter we vary is the total network size, from 100 to 6000 nodes, while keeping the percentage of overlay nodes at a constant 10%, a value which we believe is realistic. Efficiency is shown in Figure 9 and accuracy in Figure 10.

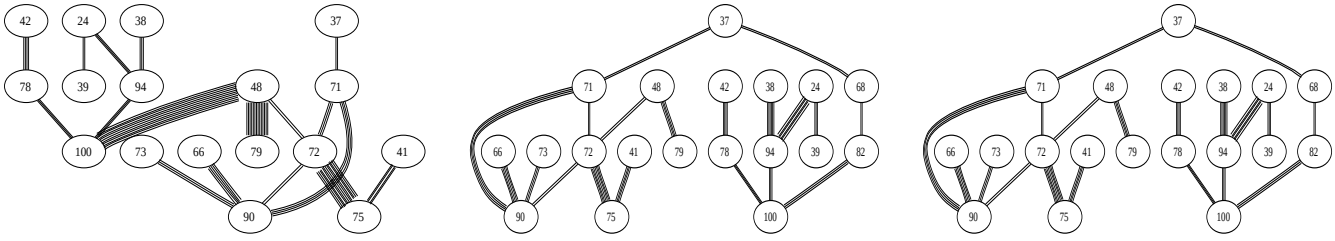
In this experiment, the efficiency and accuracy graphs are almost indistinguishable. The efficiency of Complete-LCC is almost always 1, which indicates absolute optimality; in fact, an efficiency of 1 implies that the optimum matches the optimum in the independent graph with no link correlations. Node-LCC efficiency is also optimal for almost all the network sizes, and when it is not, it is within 20% of the Complete-LCC efficiency. Although the Independent efficiency achieves the optimal for a few of the network sizes, it generally demonstrates poor performance for most sizes.

Remark. Why does the optimum so ubiquitously coincide with an upper bound that is not guaranteed to be tight, *i.e.*, the independent optimum? It is because for every overlay node u, there is an incident edge that has the maximum (independent or unicast) bandwidth b_u among all its incident edges. It is not hard to see that the bandwidth of any independent multicast tree is bounded by $\min\{b_u : \forall u\}$. Therefore, if there is sufficient capacity in the underlying network, and moreover, if the information of link correlations is effectively used to minimize stress of underlying links, it is highly likely to find a multicast tree of the independent optimal bandwidth. Clearly, in our simulation, the relative small overlay percentage leads to sufficient capacity. This is consistent with reality, since in real networks, an overlay usually resides on top of a dense underlying network whose available capacity is sufficient to serve a single overlay tree.

It can be concluded from the good performance (consistently optimal or near-optimal) of both Complete- and Node-LCC that, the heuristics algorithm *HMTC* demonstrates effectiveness in exploiting link correlation information (in the form of LCC) to obtain optimal trees. In contrast, the Independent overlay rarely exhibits good performance because it ignores link correlations, leading to inaccuracy of representing the real network and hence cannot be consistent in finding the optimal tree. The extreme similarity between respective efficiency and accuracy curves implies that it is crucial to strive for accuracy by explicitly taking into consideration link correlations.

C. Stress

To further investigate the underlying cause of poor performance of the Independent overlay and the good performance of LCC-overlays, we visually examine low-level link stress —its severity and distribution —imposed by the trees for Independent, Complete-LCC and Node-LCC.



(a) Stress graph for Independent, network size = 100.

(b) Stress graph for Complete-LCC, network size = 100.

(c) Stress graph for Node-LCC, network size = 100.

Fig. 11. Stress graphs, network size = 100.

With a fixed network size and overlay-to-low-level ratio, for each of the three trees (Independent, Complete- and Node-LCC), its constituent (overlay) links are mapped to the low-level and the stress placed by them on all the low-level edges are visually presented in the stress graph. Recall that the stress graph contains only low-level edges

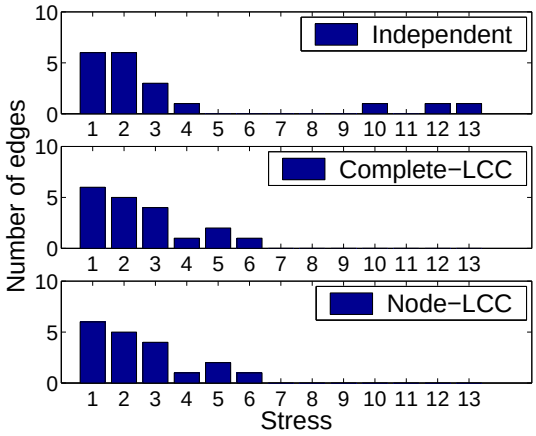


Fig. 12. Distributions of link stress for Independent, Complete-LCC and Node-LCC, network size = 100, 10% overlay nodes.

with at least one overlay tree link mapped to it. For every low-level edge (u, v) with stress s, in the stress graph, s multiple edges are drawn between node u and node v. Recall that stress as defined above in Sec. V-A takes on integer values greater than or equal to 1. In this manner, we are able to visualize link stress at a glance. Higher link stress easily shows in more multiple edges or a thicker line between two nodes.

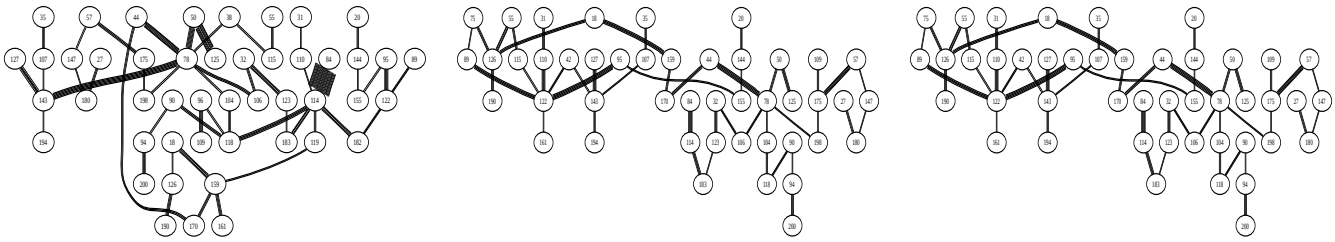
Consider the scenario of network size 100 and overlay ratio 10%; it corresponds to the 100 point on the x-axis of the efficiency graph in Figure 9. We have shown the graph in Fig. 11 in the previous section, Sec. IV-C, for the purpose of illustrating how *HMTC* reduces stress. We include it here again, for completeness and clarity of presentation.

The Independent stress graph of Figure 11(a) clearly shows three thick trunks, indicating heavy stress. This is confirmed by the distribution of stress shown in the top graph of Figure 12: three links have higher stress. The stress graphs of Complete-LCC and Node-LCC are in Figure 11(b) and (c), respectively. In this particular scenario, they look the same, containing only relatively thin lines between nodes, *i.e.*, links have relatively low stress. Their distributions of stress, bottom two graphs in Figure. 12, show that the maximum stress is half of that for Independent.

Herein lies the reason for optimal efficiency of Complete-LCC and Node-LCC and for poor efficiency of Independent, as seen in Figure 9 at the value of 100 on x-axis (the first point on the curves). The maximum link stress placed by the Independent tree is twice as high as the maximum link stress caused by the two LCC trees. Incidentally, the fact that Independent efficiency is roughly half of that of the two LCC is a convincing indication that our definition of stress is good in representing the load placed on a link relative to its capacity.

We observe that the virtue of *HMTC* with LCC is its success in choosing overlay links that spread load evenly among the low-level links, minimizing stress placed on any single link. The success is prominent even with the

much restricted Node-LCC.



(a) Stress graph for Independent, network size = 200.

(b) Stress graph for Complete-LCC, network size = 200.

(c) Stress graph for Node-LCC, network size = 200.

Fig. 13. Stress graphs, network size = 200.

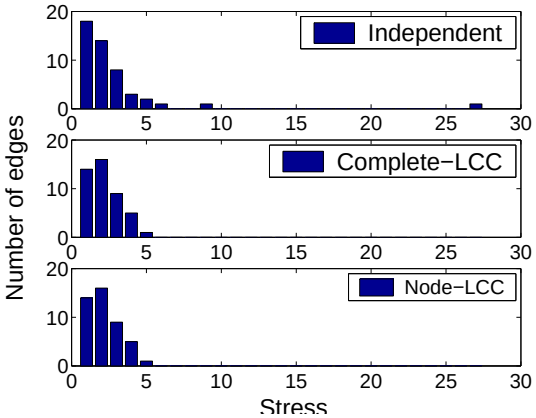


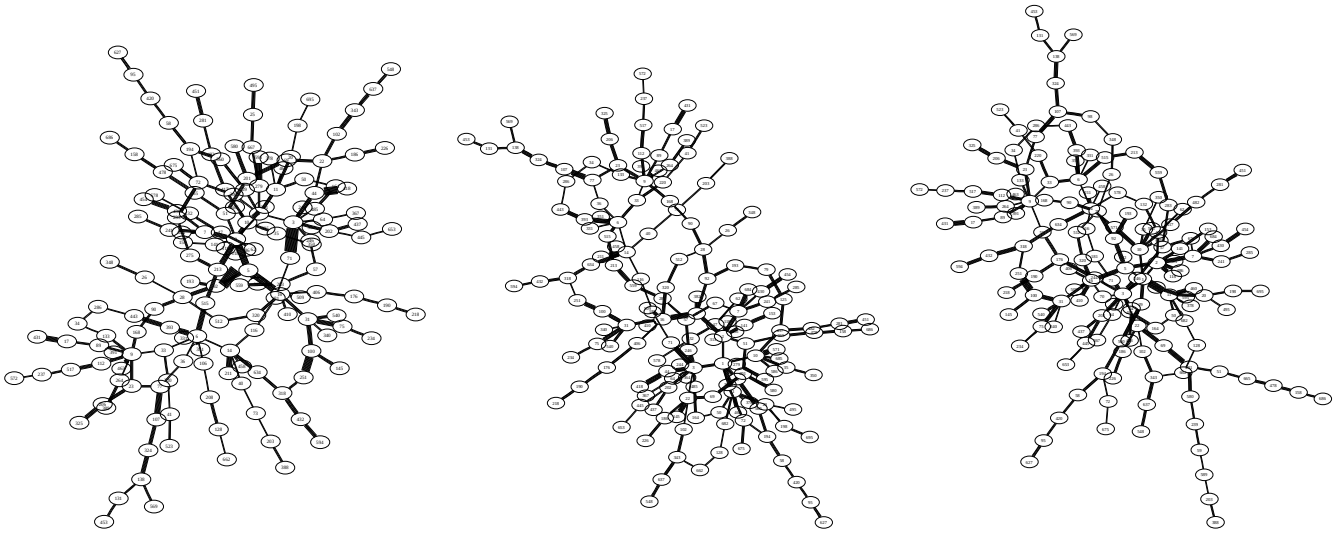
Fig. 14. Distributions of link stress in Independent, Complete-LCC and Node-LCC, network size = 200, 10% overlay nodes.

In a different scenario of 200 network nodes, 10% of which are overlay nodes, the three stress graphs are given in Figure 13. It is obvious that a low-level link in the Independent stress graph, in Figure 13(a), has much higher stress than all the rest —visually, it is an extra thick trunk in the right part of the graph. The distribution of stress in the top graph of Figure 14 shows that there is a link with stress greater than 25, whereas maximum stress for both LCC's is 5. Correspondingly, the Independent efficiency is less than $1/5$ of the efficiency of both Complete-LCC and Node-LCC, as given by the second point on the curves in Figure 9.

We examine a third scenario where the efficiency of all three trees are different, where network size is 700 with 10% overlay nodes; it corresponds to the 7th point on the efficiency curves in Figure 9. The stress graphs in Figure 15 still show Node-LCC and Complete-LCC to be similar, while stress for Independent is clearly higher in a few links. Figure 16 informs that the highest stress for Node-LCC is between those for Complete-LCC and Independent, leaning more to Complete-LCC —this is reflected in their respective efficiencies.

D. Convergence and Scalability

We analyze the convergence of our algorithm *HMTC* in Figure 17. For a fixed network of 300 nodes and increasing overlay ratios, the (normalized) tree bandwidth attained is plotted against the number of rounds that has



(a) Stress graph for Independent, network size = 700.

(b) Stress graph for Complete-LCC, network size = 700.

(c) Stress graph for Node-LCC, network size = 700.

Fig. 15. Stress graphs, network size = 700.

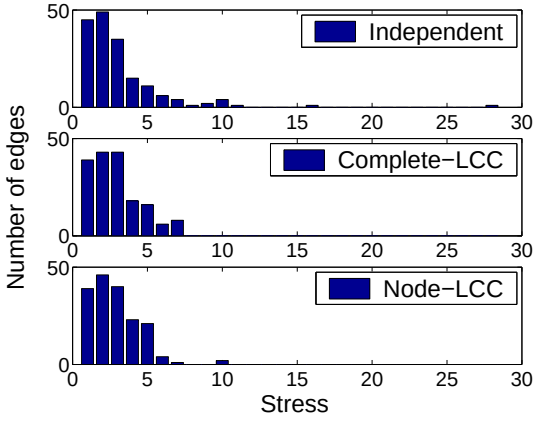


Fig. 16. Distributions of link stress for Independent, Complete-LCC and Node-LCC, network size = 700, 10% overlay nodes.

been executed in the algorithm. At each round, a link is selected to be potentially replaced; it is replaced only if another link can be found to improve the bandwidth. This is why there are instances where even though the number of rounds has increased, the tree bandwidth does not improve.

It can be seen from the convergence graph that the final bandwidth is always reached in a small number of rounds, for all overlay ratios. Although the number of rounds required to converge increases slightly for higher overlay ratios, it remains small. We do not show the cases of higher overlay ratios to maintain clarity in the graph, but for all ratios up to 80%, the algorithm converges in less than 50 rounds. Scalability of the algorithm is supported by the fact that similar convergence occurs for network sizes up to the high thousands. Moreover, in our simulations, we set the maximum number of rounds to be 100. Hence, the algorithm never runs for more than 100 rounds and still yields the near-optimal efficiency for all network sizes.

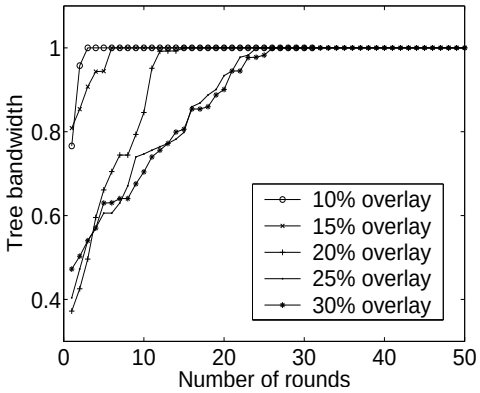


Fig. 17. Convergence of heuristics algorithm for various overlay ratios, low-level size = 300.

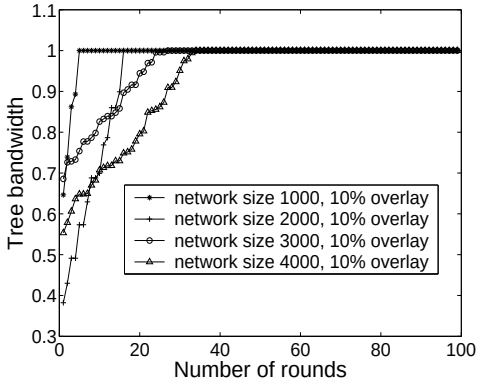


Fig. 18. Convergence of heuristics algorithm for various network sizes with 10% overlay nodes.

VI. Distributed Multicast in LCC-Overlay

A salient conclusion drawn from the evaluation of the algorithm *HMTC* in the previous section, Sec. V, is that *HMTC* is able to find near-optimal multicast trees with only node-based LCC, not the complete LCC. The reason we studied the restricted LCC class of node-based LCC is precisely because they are inherently distributed.

Two obstacles must be overcome to realize a distributed algorithm for finding a multicast tree in LCC-overlays: (1) How do we practically construct an LCC-overlay in a decentralized fashion? (2) The algorithm *HMTC* must be modified to become fully distributed.

We have previously proposed, in [12], a decentralized scheme for constructing an LCC-overlay. In this scheme, every overlay node strategically probes to obtain its own set of node-based LCC. That is, for each node, its set of LCC contains only capacity variables of its incident links. Because this is out of the scope of this paper and also due to space constraint, we do not present the details of our LCC-overlay construction scheme. To summarize, it is based on an existing efficient technique for detecting shared bottlenecks, proposed by Katabi et al. in [17], [18]. In our scheme, the node-based LCC are obtained in iterations of increasing refinement. Our simulation results showed that nearly complete node-based LCC can be discovered in a small number of refinement stages.

Once the LCC-overlay is constructed and maintained, one needs only to develop a distributed version of *HMTC*

using node-based LCC. The distributed algorithm differs from the centralized version in two aspects. In the centralized algorithm, global link replacement is done — the worst link is replaced by another that improves the tree bandwidth. In the distributed algorithm, every node does local replacement of its worst incident link in the tree, by one of its incident links in the overlay; the replacement is done whenever the new link has a higher bandwidth allocated by the LCC (not only when bandwidth of the entire tree is increased).

The other difference is the new mechanism for cycle avoidance in the distributed algorithm. When nodes make local link replacements, cycles may be introduced and it would no longer be a tree. In order to preserve the tree structure, whenever a node u is beginning the process of replacing a local link, a "replacing" notification is passed down to all nodes in the subtree rooted at u; when u is finished, a "finished" notification is passed down. When a node v receives a "replacing" notification from its parent, v forwards it to its children, and *does not accept any requests to add new links incident to it* by other nodes who may be doing their own local adjustments —until v receives the "finished" notification.

Simulation results in Figure 19 show that the distributed algorithm based on node-based LCC converges to the optimal or near-optimal, in less than 100 rounds for relatively large networks.

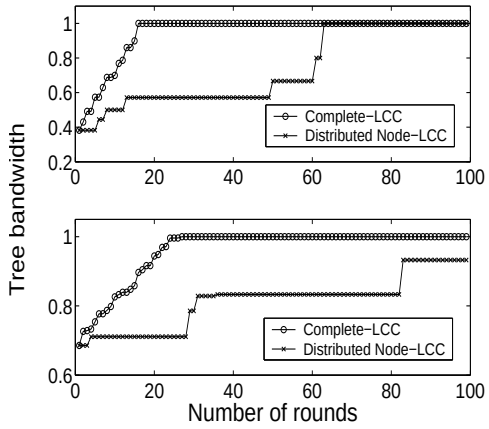


Fig. 19. Top graph: Convergence of centralized Complete-LCC and Distributed Node-LCC, for network size 1000, 10% overlay. Bottom graph: Convergence of centralized Complete-LCC and Distributed Node-LCC, for network size 4000, 10% overlay.

VII. Conclusions

In this paper, we have studied the problem of high-bandwidth overlay multicast in overlay networks with linear capacity constraints. We showed that the problem is NP-complete and proposed a heuristics algorithm. Extensive simulation results demonstrated that the achievable bandwidth of any multicast tree found in the independent overlay (*i.e.*, traditional overlay model, with no link correlations) is often much lower than the optimal. In contrast, our heuristics algorithm is able to attain near-optimal (often optimal) performance using LCC, even with node-based

LCC which is a restricted class of LCC that is naturally distributed. The evaluation of the algorithm indicates that it converges quickly and is scalable. We also proposed a distributed way of constructing high-bandwidth multicast trees with node-based LCC. We are working on the implementation of LCC-overlays and obtaining optimal multicast topologies in them. Future work also includes adaptively inferring LCCs in dynamic network conditions and multicast topologies beyond a single tree.

REFERENCES

[1] Y. Chu, S. G. Rao, S. Seshan, and H. Zhang, "A Case for End System Multicast," *IEEE Journal on Selected Areas in Communications*, pp. 1456–1471, October 2002.

[2] S. Banerjee, B. Bhattacharjee, and C. Kommareddy, "Scalable Application Layer Multicast," in *Proc. of ACM SIGCOMM*, August 2002.

[3] M. Castro, P. Druschel, A.-M. Kermarrec, A. Nandi, A. Rowstron, and A. Singh, "SplitStream: High-Bandwidth Multicast in Cooperative Environments," in *Proc. of the 19th ACM Symposium on Operating Systems Principles (SOSP 2003)*, October 2003.

[4] S. Ratnasamy, M. Handley, R. Karp, and S. Shenker, "Topologically-Aware Overlay Construction and Server Selection," in *Proc. of the IEEE INFOCOM*, 2002.

[5] M. Kwon and S. Fahmy, "Path-aware Overlay Multicast," *Computer Networks*, vol. 47, no. 1, pp. 23–45, January 2005.

[6] I. Stoica, R. Morris, M. F. Kaashoek, and H. Balakrishnan, "Chord: A scalable peer-to-peer lookup service for internet applications," in *Proc. of ACM SIGCOMM*, 2001.

[7] A. Young, J. Chen, Z. Ma, A. Krishnamurthy, L. Peterson, and R. Wang, "Overlay Mesh Construction Using Interleaved Spanning Trees," in *Proc. of INFOCOM*, 2004.

[8] K. Shen, "Structure Management for Scalable Overlay Service Construction," in *Proc. of NSDI*, 2004.

[9] J. Byers and J. Considine, "Informed Content Delivery Across Adaptive Overlay Networks," in *Proc. of ACM SIGCOMM*, August 2002.

[10] D. Kostic, A. Rodriguez, J. Albrecht, and A. Vahdat, "Bullet: High Bandwidth Data Dissemination Using an Overlay Mesh," in *Proc. of ACM SOSP*, 2003.

[11] V. Padmanabhan, H. Wang, P. Chou, and K. Sripanidkulchai, "Distributing Streaming Media Content Using Cooperative Networking," in *Proc. of the 12th International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV 2002)*, Miami Beach, Florida, May 2002.

[12] Y. Zhu and B. Li, "Overlay Networks with Linear Capacity Constraints," in *to appear in the Proceedings of the Thirteenth IEEE International Workshop on Quality of Service (IWQoS 2005)*, 2005.

[13] M.S. Kim, Y. Li, and S.S. Lam, "Eliminating Bottlenecks in Overlay Multicast," in *Networking 2005*, 2005.

[14] M.S. Garey and D.S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W. H. Freeman, New York, 1979.

[15] A. Medina, A. Lakhina, I. Matta, and J. Byers, *BRITE: Boston University Representative Internet Topology Generator*, http://www.cs.bu.edu/brite.

[16] C. Faloutsos, M. Faloutsos, and P. Faloutsos, “On Power-Law Relationships of the Internet Topology,” in *Proc. of ACM SIGCOMM*, August 1999.

[17] D. Katabi and C. Blake, “Inferring Congestion Sharing and Path Characteristics from Packet Interarrival Times,” Tech. Rep., Laboratory of Computer Science, Massachusetts Institute of Technology, 2001.

[18] D. Katabi, I. Bazzi, and X. Yang, “A passive approach for detecting shared bottlenecks,” in *Proc. of ICCCN ’01*, 2001.