Bandwidth Prediction in the Face of Asymmetry*

Sven Schober[1], Stefan Brenner[2], Rüdiger Kapitza[2], and Franz J. Hauck[1]

[1] Institute of Distributed Systems, University of Ulm, Germany,
{sven.schober,franz.hauck}@uni-ulm.de
[2] TU Braunschweig, Germany
{brenner,rrkapitz}@ibr.cs.tu-bs.de

Abstract. An increasing number of networked applications, like video conference and video-on-demand, benefit from knowledge about Internet path measures like available bandwidth. Server selection and placement of infrastructure nodes based on accurate information about network conditions help to improve the quality-of-service of these systems. Acquiring this knowledge usually requires fully-meshed ad-hoc measurements. These, however, introduce a large overhead and a possible delay in communication establishment. Thus, prediction-based approaches like Sequoia have been proposed, which treat path properties as a semimetric and embed them onto trees, leveraging labelling schemes to predict distances between hosts not measured before. In this paper, we identify asymmetry as a cause of serious distortion in these systems causing inaccurate prediction. We study the impact of asymmetric network conditions on the accuracy of existing tree-embedding approaches, and present *direction-aware embedding*, a novel scheme that separates upstream from downstream properties of hosts and significantly improves the prediction accuracy for highly asymmetric datasets. This is achieved by embedding nodes for each direction separately and constraining the distance calculation to inversely labelled nodes. We evaluate the effectiveness and trade-offs of our approach using synthetic as well as real-world datasets.

Keywords: Asymmetric bandwidth prediction, tree embedding

1 Introduction

The performance of distributed multimedia applications largely depends on path properties like latency, packet-loss and bandwidth. A priori knowledge of these helps to improve the user-perceived quality of service by the adaption of application-specific variation points, like video resolution and codec, or modifying the communication structure by placing infrastructure nodes at beneficial locations in the network.

* The original publication is available at www.springerlink.com

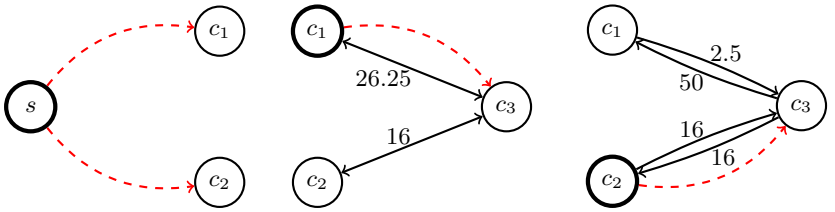
A naïve approach for acquiring path-property knowledge is to perform ad-hoc measurements. However, this has several disadvantages. First, it creates a huge overhead, as the required measurements grow quadratically with the node count. Second, it introduces delay as measurements cannot be performed in parallel, but have to be performed sequentially to avoid interferences.

Therefore, multiple prediction-based approaches have been proposed [1, 4, 5, 8, 10, 16, 18, 21, 25] which reduce the amount of required measurements by embedding hosts into a metric space and predicting unknown inter-host path properties using a distance function on the host's coordinates. Although predicting latency has been well studied in the literature, fewer approaches exist which are able to predict path bandwidth [8, 10, 16].[3] Tree metric spaces also have been proposed as targets for embedding network path metrics [1, 21, 25].

Most of these approaches require a symmetric distribution of the metric under consideration. However, this assumption is often violated [11, 19], the most obvious example being the last-mile link of an endhost, often implemented with access technologies like ADSL[4] and DOCSIS[5] (Cable). Consequently, matrix factorization has been proposed to cope with asymmetry but needs clustering of nodes to reduce the rank of the distance matrices. The model proposed by Beaumont et al. [2] is also able to cope with asymmetry, but assumes the access link to determine the bottleneck bandwidth of each path. Xing et al. [27] propose embedding bandwidth in a set of ultrametric spaces, but their system is based on landmarks, and thus, not fully decentralizable.

To further motivate the need to cope with path asymmetry consider a *peer assisted streaming* system [14]. These systems try to minimize server load by selecting "close-by" peers as streaming sources. Figure 1a depicts a session between a server s and two clients c_1 and c_2 (active sources are rendered bold). Consider a scenario where c_1 has a highly asymmetric access link of 50 Mbit/s downstream and 2.5 Mbit/s upstream, common values for cable Internet, and c_2 has a symmetric though lower bandwidth link of 16 Mbit/s in each direction. We further assume for simplicity, that there is no bottleneck on the path between s and the clients. A new client c_3 joining this session has three choices s, c_1 and c_2 as its streaming source. Assuming s is already highly loaded, joining clients are forced to select other sources for streaming. In a system leveraging a symmetric bandwidth prediction component the choice will be node c_1 as this system averages up- and downstream in its prediction (cf. Figure 1b black arrows). However, c_2 obviously would be a better choice. Moreover, assuming the stream consumes bandwidth greater than 2.5 Mbit/s, selecting c_2 would lead to unacceptable performance of the streaming system. An asymmetry-aware streaming system would be able to select a peer based on its upstream bandwidth and in the presented case, select c_2 as its stream source (cf. Figure 1c).

[3] Note that this dicussion is independent of the type of bandwidth under consideration: capacity or available bandwidth.
[4] ITU-T G.992.5
[5] ITU-T J.222



(a) Streaming Session (b) Symmetric Peer Selection (c) Asymmetry-aware Peer Selection

Fig. 1: Peer Selection Alternatives c_1 and c_2 in a Peer-assisted Streaming Scenario

In this paper we extensively study the impact of asymmetry on existing prediction approaches and present a technique, *direction-aware embedding* (DAE), effectively mitigating the negative effects of asymmetry on the prediction accuracy. By separating the upstream and downstream path characteristics and embedding direction-labelled nodes onto prediction trees, we are able to significantly improve the prediction performance of previously presented tree-embedding schemes.

The rest of this paper is organized as follows. First, we present existing tree-embedding schemes in Section 2. Next, we present our approach in Section 3, which tackles this problem by applying our direction-aware embedding scheme. In Section 4 we evaluate the negative impact asymmetry has on the prediction accuracy of previous systems and the effectiveness of our novel approach. Finally, we discuss related work in Section 5 and conclude in the last section.

2 Background

Before we can present our contribution we first discuss the basic concepts of tree metrics, distance labeling and how both are applied in the two existing systems *Sequoia* [21] and its improved version by Song et al. [25].

In the scope of this paper, a *host* refers to a computer or Internet host, while a *node* denotes the respective entry on a tree. We also differentiate estimation and prediction. While we use the term *estimation* for sophisticated measurement techniques like pathChirp [22], the term *prediction* refers to reading tree distances off an existing prediction tree without further measurements.

2.1 Tree Metrics and Bandwidth

Following Ramasubramanian et al. [21] we understand a set of pairwise bandwidth measurements M as a tree metric, if there exists a tree T representing the measurements as distances between tree nodes with non-negative edge-weights such that $M \subseteq T$ and $d_M(a,b) = d_T(a,b)$ for all $a, b \in M$. While $d_M(a,b)$ represents the measured bandwidth from host a to b, $d_T(a,b)$ denotes the predicted bandwidth and accordingly the distance between the nodes on the tree.

In analogy to the triangle inequality, a necessary condition for a metric to be embeddable on a tree is the four-points condition (4PC): $d(w,y) + d(x,z) = d(w,z) + d(x,y)$ for the distances between four nodes w, x, y, z ordered by renaming such that $d(w,x) + d(y,z) \leq d(w,y) + d(x,z) \leq d(w,z) + d(x,y)$. Said in words, the two greater sums of distances between the nodes have to be equal, to embed them on a tree without distortion.

The key observation the authors of Sequoia make is that distance matrices as perceived in the Internet display a certain *treeness*. More formally, they satisfy a relaxed form of the 4PC, the so called ϵ-four-points condition (ϵ-4PC) for relatively small ϵ-values:

$$d(w,z) + d(x,y) \leq d(w,y) + d(x,z) + 2\epsilon \cdot min\{d(w,x), d(y,z)\} \tag{1}$$

Here, $\epsilon \in [0,1]$ characterizes how close a given metric is to a tree-metric ($\epsilon = 0$ is absolute treeness). Ramasubramanian et al. depict that bandwidth measurements on PlanetLab [3] show a high degree of treeness, with 80% of ϵ-values being less than 0.2.

The basic assumption of previous embedding approaches is that the source metric is at least a *semimetric*, i.e. it is assumed to be symmetric. However, in real-world network topologies this assumption is often violated [19, 11]. Following Xing et al. [27], we define an asymmetry coefficient ζ_{ab}, which denotes the bandwidth asymmetry between two nodes a and b as follows:

$$\zeta_{ab} = \frac{|(d(a,b) - d(b,a)|}{d(a,b) + d(b,a)} \tag{2}$$

2.2 Sequoia

Ramasubramanian et al. present *Sequoia* [21], a system which is able to predict latency and bandwidth between its participants. It reduces the total number of measurements by embedding hosts on an edge-weighted tree, called *prediction tree*. Then, distances between arbitrary hosts can be read off that tree, by summing up the weight of the shortest path between them.

As bandwidth is not an additive but a concave metric,[6] a transformation has to be applied to the measured values. This is done for Sequoia in a linear fashion by subtracting the measured values from a large constant.

The prediction tree is constructed using an embedding procedure that selects two nodes (anchor and lever) from the existing tree by maximizing the so called *Gromov product* individually for each joining host. Concrete, the Gromov product $(b|c)_a$ allows the calculation of the intersection points of the incircle with the edges of a triangle Δ_{abc}:

$$(b|c)_a = \frac{1}{2}(d(b,a) + d(c,a) - d(b,c)) \tag{3}$$

[6] Path-bandwidth is defined by the weakest link of a path, while path-latency is the sum of individual link latency

In the context of prediction trees, b depicts the anchor, a the lever or "base node" and c the host to be embedded. Note, that the distances involving c have to be measured, whereas $d(b, a)$ can be read off the tree. Once anchor and base have been found, a virtual node t_c is inserted along the path p_{ba} at distance $(b|c)_a$ from the base node. Figure 2a displays a sample prediction tree after four node insertions.

A special *distance-labelling* scheme [20] for the nodes of a prediction tree allows the encoding of distances between any two hosts of an edge-weighted tree inside their respective labels. This enables the computation of distances between two hosts without knowledge of the complete prediction tree. In the context of this work, we follow the labelling scheme used by Song et al., where labels consist of a list of 3-tuples representing the anchor hierarchy and the distances between the nodes:

$$l_v = (a_0, d(a_0, t_v), d(t_v, v)), ... \tag{4}$$

Following this scheme we get the following distance labels for the nodes in the example tree depicted in Figure 2a: $l_a = (a, 0, 0)$, $l_b = (a, 0, 41)(b, 0, 0)$, $l_c = (a, 0, 41)(b, 3, 4)(c, 0, 0)$ and $l_d = (a, 0, 41)(b, 16, 13)(d, 0, 0)$. Node c for example is embedded on the tree using b as its anchor and its virtual node being at offset 3 on the path from b to a. Then, c is connected to its virtual node at distance 4.

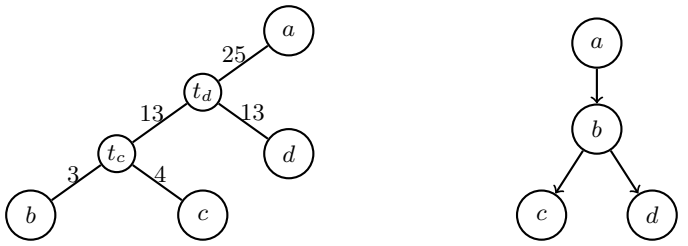
The first node of a tree can easily be identified by a label consisting of only one 3-tuple. For the other nodes the last tuple represents the node itself (with offset and distance = 0 because there is no distance to itself) while the other tuples represent distances to its anchor, its anchor's anchor and so forth. Given their labels, the tree distance between two hosts can be calculated easily [24]. Basically, considering $d_T(c, d)$ we get: $16 - 3 + 4 + 13 = 30$ for example.

In order to reduce the amount of conducted measurements the authors introduce an abstraction called *anchor tree*, wherein they capture the anchor relationships of joining nodes (which are also manifested in the distance labels). Figure 2b shows the anchor tree corresponding to the presented prediction tree. Since anchor trees must contain distance labels, an anchor tree can be transformed to a prediction tree and vice versa. Instead of searching the complete prediction tree for a node maximizing the Gromov product, the anchor search is guided by the anchor tree starting with the lever. A greedy search algorithm stops once no more progress can be made. Note, that this can result in a suboptimal anchor selection, as only a subset of anchor candidates is considered depending on the algorithm's chosen path through the anchor tree.

The embedding order of hosts influences the overall distortion e_r, defined as the mean value of the relative prediction error for each path on the tree. The relative prediction error itself is defined as follows:

$$e_p = \frac{|d_T(a, b) - d_M(a, b)|}{d_M(a, b)} \tag{5}$$

The problem of prediction accuracy being influenced by the order of embedding the hosts has been addressed by the authors of Sequoia by constructing multi-



(a) Prediction Tree (b) Anchor Tree

Fig. 2: Sample Instances of Core Sequoia Concepts

ple prediction trees in parallel using different insertion orders. This way, when predicting distances they are able to remove outliers by choosing the median of distances predicted by the constructed trees.

2.3 Optimizations

Song et al. [25] improve on Sequoia basically in two ways: First, they decentralize the tree construction algorithm. To achieve this, they discard the idea of using multiple trees and use the structure of a single anchor tree to form an overlay network between the participating nodes. Afterwards, decentralization of the embedding algorithm is done by allowing the use of a random base node and starting the anchor search at the chosen base node. Second, they improve tolerance for datasets with less than perfect treeness which is true for most real-world datasets. To this end they modify the anchor selection algorithm to minimize the overall prediction error instead of maximizing the Gromov product, extend the search space on the anchor tree and modify the transformation to a rational function to avoid negative values when prediction values exceed the transformation constant. Furthermore, Song et al. propose an anchor search optimization which optimizes the chosen base and anchor by another pass of error minimization based on measurements that have already been taken.

3 Direction-aware Embedding (DAE)

Obviously, Sequoia and Song's approach both are not designed to cope with asymmetric links. As these are a common phenomenon in the Internet, we identified them as a challenge for prediction accuracy. In order to cope with asymmetry, the key idea of our approach is to reflect the varying bandwidth properties of an asymmetric host by embedding it twice on the prediction tree. Once for its upstream and once for the downstream properties (cf. Figure 3). Distance computation then is only performed and valid between inversely directed representative nodes of two hosts.

Algorithm 1 DAE Embedding Procedure

Input: j: joiner, d: direction
Output: l_j: distance-label of joiner

1: $b \leftarrow$ random node with direction $-d$
2: $a \leftarrow null$, $e_{min} \leftarrow \infty$, $v_{base} \leftarrow b$
3: **while** $a \neq v_{base}$ **do**
4: $\quad C \leftarrow$ getCandidates(v_{base}, d)
5: $\quad$ **for all** $c \leftarrow C$ **do**
6: $\quad\quad l_{a-tmp} \leftarrow$ position(j, c, b)
7: $\quad\quad e_p \leftarrow$ relativeError(l_{a-tmp})
8: $\quad\quad$ **if** $e_p < e_{min}$ **then**
9: $\quad\quad\quad e_{min} \leftarrow e_p$, $a \leftarrow c$
10: $\quad\quad$ **end if**
11: $\quad$ **end for**
12: $\quad v_{base} \leftarrow a$
13: **end while**
14: **return** l_{a-tmp}

Algorithm 2 DAE Positioning

Input: j: joiner, a: anchor, b: base
Output: l_j: new distance-label

1: $\delta_o \leftarrow d_T(a, b) - (a|j)_b$
2: $\delta_d \leftarrow d(j, b) - (a|j)_b$
3: **return** appendLabel(a, δ_o, δ_d)

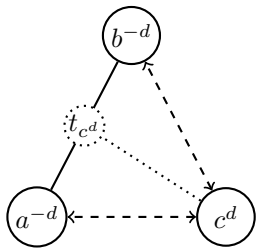


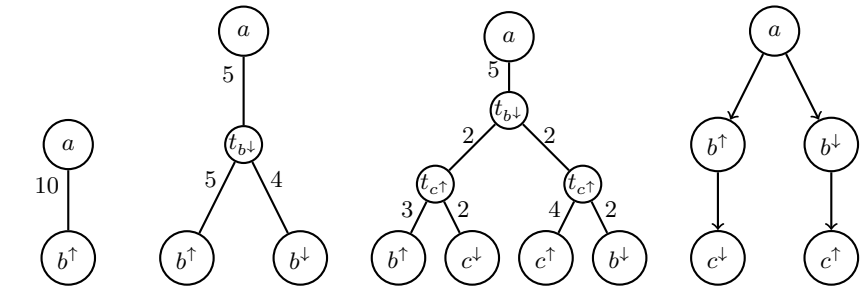
Fig. 3: Embedding Visualization

In order to embed a node on a DAE prediction tree, we modified the tree construction algorithm by first assigning a direction-label $d \in \{\uparrow, \downarrow\}$ to the host name. The embedding procedure itself then is executed twice in a similar way as Sequoia for upstream and downstream properties individually. However, in contrast to the existing approaches, the chosen base node and the anchor both have to be of inverse direction $-d$ to the joining node (cf. Figure 3 and Algorithm 1). Hence, a single host is represented by two nodes on the tree making use of a total of four different reference nodes (an anchor and a base node for each direction). For the upstream base and anchor, the corresponding upstream bandwidth starting from the joiner is measured and used as a distance for the embedding algorithm. The same is done for the opposite direction using the appropriate downstream measurements from the base and anchor to the joiner. This may result in the two nodes of a single host residing in totally different sections of the tree.

Our embedding procedure is depicted in Algorithm 1. Here, j denotes a joining node, a an anchor candidate, b the base node and d a direction-label. We choose a base node randomly and select an anchor using the anchor tree. Due to the inverse direction constraint, the anchor tree is a bipartite graph consisting of alternating up- and downstream nodes. Consequently, inappropriate nodes (wrong direction) have to be skipped during the anchor search. This is done by getCandidates (Line 4) which only selects valid nodes of the two-hop neighborhood[7] of the given v_{base}-node which is the starting point for each anchor search iteration. Following Song's approach, we choose a node minimizing the

[7] Two hops are needed because of the alternating directions on the anchor tree

relative prediction error e_p (Lines 5-11), as defined in Equation 5. When no anchor can be found for a given base node because of non-available measurements, we choose another base node as a fallback. We also use the optimization procedure based on already measured links as proposed by Song et al. Note that nodes of the anchor tree might also have multiple children alike Song's approach.



(a) Join $b^\uparrow$ (b) Join $b^\downarrow$ (c) Join c (d) Anchor Tree

Fig. 4: DAE Example Tree Construction

Positioning of hosts on the tree is done similar to Sequoia (c.f. Algorithm 2). The Gromov product is used to calculate the position of a node with respect to its anchor. The distances needed for this calculation are measured ($d_M(j,b), d_M(j,a)$) and read off the tree ($d_T(a,b)$).[8] Then, the according offset δ_o and distance δ_d values are calculated and the node is embedded on the tree. This is done by copying the distance label of the anchor and altering it, appending a new 3-tuple containing offset and distance with respect to the anchor. A step-wise sample prediction tree construction using our DAE approach is depicted in Figure 4 with the corresponding anchor tree presented in Figure 4d.

Special care has to be taken when embedding the first node, as it is represented only by a single node. Since it bears no direction-label, it can act as a base node for both up- and downstream representative nodes. Also the second node, which will be one direction-labelled representative of the second host, has to be treated specially. This is due to the fact that there is no node in the tree, which could act as anchor, as the first cannot be base and anchor at the same time. Thus, the weight of the edge to the first node simply represents the measured distance in the corresponding direction (c.f. Figure 4a). Afterwards, the third node (the second representative of the second host) is embedded on the edge between the two existing nodes at the measured distance (c.f. Figure 4b). For the second host one can decide whether to embed its upstream or its downstream node first. We embed the node with the longer distance to the first node before the other one, as this avoids negative distances δ_d.

[8] Note that we also use the rational transformation introduced by Song et al. in order to transform bandwidth measurements.

In order to predict the path bandwidth from host a to b in our scheme, we first assign the implied direction-labels, $a^{\uparrow}$ and $b^{\downarrow}$ and then calculate the tree distance. Note that the tree distance between two nodes of the same direction (e.g. $d(a^{\uparrow}, b^{\uparrow})$) is meaningless as is the distance between the two representatives of a single host (e.g. $d(a^{\uparrow}, a^{\downarrow})$).

4 Evaluation

In the following, the benefit and properties of DAE especially in presence of asymmetric links are shown and evaluated. First, we describe our methodology, topologies and algorithm configuration. Then, we proceed to quantify the improvement of prediction accuracy achievable by DAE. We used two different topologies described below. We embed these using the three embedding approaches Sequoia, its enhancement by Song et al. (denoted as "Song" in the figures) and our DAE algorithm. Since there were no publicly available implementations of Sequoia and Song's approach, we implemented them based on the respective papers [21, 25][9].

4.1 Methodology

When a dataset has been embedded, we calculate the relative embedding error for the complete prediction tree as the average value of individual link prediction errors. In this evaluation, the relative prediction error e_p of the path between two nodes a and b is calculated according to Equation 5. Note that the error calculation is also direction-aware.

It is also vital to define how the amount of executed measurements is counted for this evaluation. Since Sequoia and Song's approach both take the average value of the bidirectional measurements $a \to b$ and $b \to a$ between the hosts a and b, we count two measurements for each measured link. For DAE it is possible to make use of a measurement only for one direction of a link. Hence, counting both link directions individually is also appropriate for DAE.

4.2 Topologies

PlanetLab Topology: In order to compare our approach against the two existing tree-based approaches (Sequoia and Song's approach) we created this topology based on a measurement dataset between about 385 PlanetLab hosts. The snapshot[10] we used contained about 130,000 measurements acquired by the bandwidth estimation tools PathRate [6], PathChirp [22] and Spruce [26].

[9] Our implementations are available at http://www.uni-ulm.de/en/in/vs/proj/ic2

[10] Acquired 2010-08-28 at 4:57:51PT using S^3; http://networking.hpl.hp.com/s-cube

Synthetic Topology: We investigate the impact of asymmetry on the prediction accuracy of Sequoia, Song and DAE using a synthetic dataset, which allows the definition of a particular asymmetry ratio r_ζ. It is generated by creating a certain amount of hosts, defined by two parameters: "upstream" and "downstream". The downstream value for a node is set randomly, while the corresponding upstream value is calculated based on the downstream bandwidth as $r_\zeta \cdot bw_\downarrow$ for an asymmetry factor $r_\zeta \in (0, 1)$. The synthetic scenario is based on the assumption that link bandwidth only depends on the last mile of a particular path. Thus, the bandwidth between two hosts is given by the minimum value of the upstream of the sender and the downstream of the receiver of a transmission as $bw(a, b) = \min(a_\uparrow, b_\downarrow)$. Note, that as a consequence of our derivation scheme of a distance matrix from our model, we need to contaminate our synthetic topology with hosts of high and symmetric bandwidth, too.[11] Otherwise, high downstream bandwidth would not be noticeable since no upstream bandwidth is high enough. We call this "pseudo-symmetry."

4.3 Prediction Tree Algorithms

Recall, that *Sequoia* constructs multiple (k) trees to mitigate distortion due to insertion order.[12] Furthermore, when embedding the dataset using Sequoia, the bandwidth between two nodes is defined as the average value of the bidirectional measurements. In case a measurement is only available for one direction of a link we assume the other direction to be equal. Following the authors of Sequoia, we set the Gromov product to negative infinity if one of the measurements is still not available.

In our evaluation of *Song*, we only use a single tree as proposed in the corresponding paper to generate our error analysis. Furthermore, symmetrisation of measurements is also done for Song's approach.

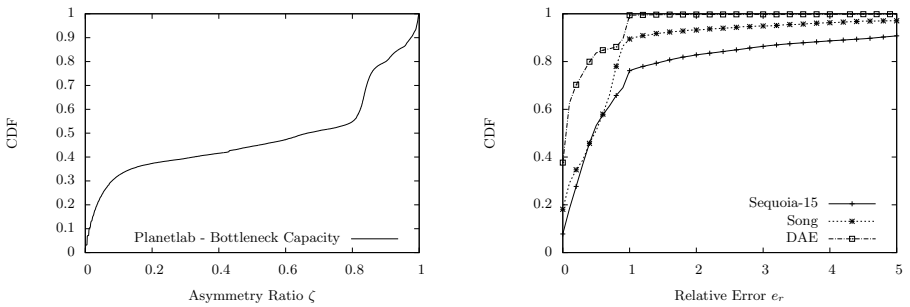
When embedding the dataset using DAE, bidirectional measurements are *not* averaged and, in contrast to the other approaches, missing measurements for only one direction are not replaced by the other direction but implicitly tolerated by our algorithm by using another anchor. Our algorithm also might make use of a unidirectional measurement value although the other direction is missing.

4.4 Accuracy of Prediction

Figure 5a depicts the cumulative distribution of ζ-values in the PlanetLab topology. It is obvious that while roughly one third of hosts feature largely symmetric link conditions (small values on the x-axis), the other two thirds exhibit high values of asymmetry. Such a strong asymmetry on PlanetLab is surprising, as we expected hosts from research institutions to have good and symmetric connections. Asymmetry might be partially explained by varying points in time when

[11] We doted our topology with 33% high-bandwith, symmetric hosts.
[12] We set $k = 15$, as this provides 15% higher accuracy than $k = 10$.

the measurements were performed. Nevertheless, we conclude that asymmetry will be even stronger in settings comprising private Internet access links.



(a) Asymmetry Distribution (b) Relative Link Prediction Error

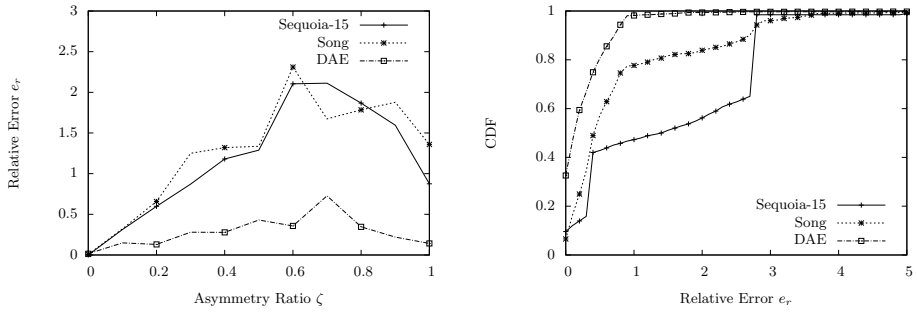
Fig. 5: PlanetLab Topology

When we feed this dataset to Sequoia and compare the predicted to the measured bandwidth, instead of the bidirectionally averaged values, we get the distribution depicted in Figure 5b. We see a heavy tail of huge prediction errors for $e_p > 1$. These would render the system practically unusable for peer selection purposes as motivated in the introduction.

As the PlanetLab dataset only represents a single mean ζ_{pl}-value, we studied the prediction performance with respect to varying ζ-values $\in [0, 1]$ using our synthetic topology. Figure 6a shows that increasing asymmetry has a severely negative impact on Sequoia's prediction performance. The advantage of DAE is proportional to the ratio and amount of asymmetry present in the dataset. There is no big advantage over Sequoia and Song for symmetric network conditions. In the presence of asymmetry, DAE explicitly takes this into account and allows a more accurate prediction.
In Figure 6a the error for various ζ-values is shown for each of the three tree algorithms. As can be seen, DAE significantly improves the accuracy of bandwidth prediction compared to Sequoia and Song. Song performs worse than Sequoia in this scenario, as it only constructs a single tree. We explain the increasing accuracy of Sequoia and Song's approach in Figure 6a for ζ-values above 0.7 by the measurement phenomenon pseudo-symmetry described in Chapter 4.2.

Considering the amount of measurements needed to construct the prediction trees, we found that Sequoia performed 84% of all $n(n-1)$ possible measurements for PlanetLab,[13] Song 13% and DAE 44%. We argue that DAE strikes a reasonable balance between accuracy and measurement traffic.

[13] The relatively high amount of measurements results from using $k = 15$, as we get only about 25% of measurements when we use $k = 1$. Furthermore, as described in Section 4.1 we count measurements for both directions of a link individually.



(a) e_p of all Tree Algorithms (b) Relative Link Prediction Error

Fig. 6: Synthetic Dataset

5 Related Work

Prediction of Internet path properties has been extensively studied. Especially the prediction of latency has been the focus of much attention in the past. Ng et al. [18] were the first to implement the idea of embedding inter-host latencies in an Euclidian space by assigning a coordinate to them, forming the research field of network coordinate systems. Their system is based on central landmark nodes to which common nodes measure their latency and use trilateration to locate themselves in the coordinate space. This approach has been decentralized in the widely known Vivaldi system [4], where Dabek et al. use a system of interconnected springs to model the location process of each node. Vivaldi's performance has been further improved by Elser at al. [7]. Common to these approaches is that the accuracy is highly susceptible to triangle inequality violations (TIVs). To tackle this problem embeddings into hyperbolic spaces [23] have been proposed. Furthermore, there is a line of research, which takes occurrences of TIVs as a hint for optimization potential, leveraging detour routing [15, 9].

Embedding latencies is straightforward as this is an additive metric. Bandwidth on the other hand is a concave metric, and thus, does not lend itself for easy embedding in Euclidean spaces, as was shown by the authors of Sequoia [21]. Thus, systems have been devised to address this problem [8, 10, 16]. As bandwidth prediction under asymmetric bandwidth distributions is an even more challenging problem, Xing et al. [27] propose PathGuru, which embeds distance matrices in several ultra-metric spaces formed by pre-deployed landmarks. Maintaining incoming and outgoing bandwidth vectors, this system can predict asymmetric bandwidth distributions. However, the need for pre-deployed landmarks is a clear disadvantage.

There is a line of research based on matrix factorization, which was proposed to tackle the problem of triangle inequality violations. Mao et al. present the IDES system [17], a landmark-based approach which assigns each host an incoming and an outgoing vector. The distance between two hosts then is cal-

culated by the scalar product of these vectors. The approach has later been decentralized by Liao et al. [13, 12]. The core assumption common to these systems is that the distance matrix is of low rank and can be represented as the product of smaller matrices. In our problem domain, small rank corresponds to clustering of nodes as nodes in a cluster will yield highly similar rows in the distance matrix. Though, tree-based approaches do not require such clustering.

Beaumont et al. study the "last-mile" model [2] first proposed by Liu et al. [14] and assume that perceived path bandwidth between hosts is solely determined by their access links. The authors consider scenarios where this is not the case as outliers, and cut them off using a percentile. Thus, they reduce potentially diverse path properties to single values for up- and downstream, which fails to capture situations where multiple hosts share a common bottleneck. In contrast, DAE allows bottlenecks to reside anywhere in the network.

6 Conclusion

In this paper, we presented *direction-aware embedding* (DAE), our approach for bandwidth prediction which is capable to cope with highly asymmetric bandwidth distributions. To the best of our knowledge, our approach is the only one simultaneously being fully decentralizable, independent of clustering and not assuming the bottleneck to reside on the last link. Opposed to existing tree-embedding approaches our scheme is able to maintain high prediction accuracy faced with asymmetric bandwidth distributions.

References

1. I. Abraham, M. Balakrishnan, F. Kuhn, D. Malkhi, V. Ramasubramanian, and K. Talwar. Reconstructing approximate tree metrics. In *Proc. of the 26th Ann. ACM Symp. on Princ. of Distr. Comp.*, pages 43–52, Aug. 2007.
2. O. Beaumont, L. Eyraud-Dubois, and Y. J. Won. Using the last-mile model as a distributed scheme for available bandwidth prediction. In *Proc. of the 17th Int. Conf. on Par. Proc.*, pages 103–116, 2011.
3. B. Chun, D. Culler, T. Roscoe, A. Bavier, L. Peterson, M. Wawrzoniak, and M. Bowman. Planetlab: an overlay testbed for broad-coverage services. *SIGCOMM Comput. Commun. Rev.*, 33(3):3–12, July 2003.
4. F. Dabek, R. Cox, F. Kaashoek, and R. Morris. Vivaldi: a decentralized network coordinate system. *SIGCOMM Comput. Commun. Rev.*, 34(4):15–26, Aug. 2004.
5. B. Donnet, B. Gueye, and M. Kaafar. A survey on network coordinates systems, design, and security. *IEEE Comm. Surveys Tutorials*, 12(4):488–503, quarter 2010.
6. C. Dovrolis, P. Ramanathan, and D. Moore. Packet-dispersion techniques and a capacity-estimation methodology. *Trans. on Netw.*, 12(6):963–977, Dec. 2004.
7. B. Elser, A. Frschler, and T. Fuhrmann. Tuning vivaldi: Achieving increased accuracy and stability. In T. Spyropoulos and K. Hummel, editors, *Self-Organizing Systems*, volume 5918 of *LNCS*, pages 174–184. Springer, 2009.
8. P. Francis, S. Jamin, C. Jin, Y. Jin, D. Raz, Y. Shavitt, and L. Zhang. Idmaps: a global internet host distance estimation service. *Trans. on Netw.*, 9(5):525–540, Oct. 2001.

9. T. Haddow, S. W. Ho, J. Ledlie, C. Lumezanu, M. Draief, and P. Pietzuch. On the feasibility of bandwidth detouring. In *Proc. of the 12th Int. Conf. on Pass. and Act. Meas.*, PAM'11, pages 81–91, 2011.
10. N. Hu and P. Steenkiste. Exploiting internet route sharing for large scale available bandwidth estimation. In *Proc. of the 5th ACM SIGCOMM Conf. on Internet Meas.*, pages 16–16, 2005.
11. K. Lakshminarayanan and V. N. Padmanabhan. Some findings on the network performance of broadband hosts. In *Proc. of the 3rd ACM SIGCOMM Conf. on Internet Meas.*, pages 45–50, 2003.
12. Y. Liao, W. Du, P. Geurts, and G. Leduc. Dmfsgd: A decentralized matrix factorization algorithm for network distance prediction. *Trans. on Netw.*, PP(99):1, 2012.
13. Y. Liao, P. Geurts, and G. Leduc. Network distance prediction based on decentralized matrix factorization. In *Proc. of the 9th IFIP TC 6 Int. Conf. on Netw.*, pages 15–26, 2010.
14. S. Liu, R. Zhang-Shen, W. Jiang, J. Rexford, and M. Chiang. Performance bounds for peer-assisted live streaming. *SIGMETRICS Perform. Eval. Rev.*, 36(1):313–324, June 2008.
15. C. Lumezanu, R. Baden, D. Levin, N. Spring, and B. Bhattacharjee. Symbiotic relationships in internet routing overlays. In *Proc. of the 6th USENIX Symp. on Netw. Sys. Des. and Impl.*, NSDI'09, pages 467–480, 2009.
16. H. V. Madhyastha, T. Isdal, M. Piatek, C. Dixon, T. Anderson, A. Krishnamurthy, and A. Venkataramani. iplane: an information plane for distributed services. In *Proc. of the 7th OSDI Conf.*, pages 367–380, 2006.
17. Y. Mao, L. Saul, and J. Smith. Ides: An internet distance estimation service for large networks. *J. on Sel. Areas in Comm.*, 24(12):2273–2284, Dec. 2006.
18. T. S. E. Ng and H. Zhang. Predicting internet network distance with coordinates-based approaches. In *Proc. IEEE INFOCOM*, volume 1, pages 170–179, 2002.
19. V. Paxson. End-to-end routing behavior in the internet. *SIGCOMM Comput. Commun. Rev.*, 36(5):41–56, Oct. 2006.
20. D. Peleg. Proximity-preserving labeling schemes. *J. Graph Theory*, 33(3):167–176, Mar. 2000.
21. V. Ramasubramanian, D. Malkhi, F. Kuhn, M. Balakrishnan, A. Gupta, and A. Akella. On the treeness of internet latency and bandwidth. In *Proc. of the 11th Int. Joint Conf. on Meas. and Modeling of Comp. Sys.*, pages 61–72, 2009.
22. V. Ribeiro, R. Riedi, R. Baraniuk, J. Navratil, and L. Cottrell. pathchirp: Efficient available bandwidth estimation for network paths. In *Passive and Active Meas. Workshop*, 2003.
23. Y. Shavitt and T. Tankel. Hyperbolic embedding of internet graph for distance estimation and overlay construction. *Trans. on Netw.*, 16(1):25–36, Feb. 2008.
24. S. Song. Decentralized pairwise bandwidth prediction, Unknown Date. http://www.cs.umd.edu/Grad/scholarlypapers/papers/SukhyunSong.pdf.
25. S. Song, P. Keleher, B. Bhattacharjee, and A. Sussman. Decentralized, accurate, and low-cost network bandwidth prediction. In *Proc. IEEE INFOCOM*, pages 6–10, Apr. 2011.
26. J. Strauss, D. Katabi, and F. Kaashoek. A measurement study of available bandwidth estimation tools. In *Proc. of the 3rd ACM SIGCOMM Conf. on Internet Meas.*, pages 39–44, 2003.
27. C. Xing, M. Chen, and L. Yang. Predicting available bandwidth of internet path with ultra metric space-based approaches. In *Proc. of IEEE GLOBECOM*, pages 1–6, 2009.