CHESS: An Application-aware Space for Enhanced Scalable Services in Overlay Networks

Mohammad Malli, Chadi Barakat, Walid Dabbous
INRIA-Sophia Antipolis, France
{mmalli, cbarakat, dabbous}@sophia.inria.fr

***Abstract*— We introduce in this paper *CHESS*, an application-aware space for enhanced scalable services in overlay networks. In this new space, the proximity of peers is determined according to a utility function that considers the network parameters (e.g., delay, bandwidth, and loss rate) impacting application performance. We motivate the need for this new notion by showing that the proximity in the delay space does not automatically lead to a proximity in another space (e.g., space of the bandwidth). For determining the proximity in CHESS, network parameters must be estimated easily and scalably. Therefore, we use the matrix factorization approach for estimating the delay and loss parameters. Besides, we propose a scalable model that estimates the bandwidth among peers using the bandwidth of the indirect paths that join them via a set of landmarks[1]. Our idea is that an indirect path shares the same tight link with the direct path with a probability that depends on the location of the corresponding landmark with respect to the direct path or any of the two peers subject to bandwidth inference. The results show that characterizing the proximity in CHESS provides a much better quality than that obtained when using the delay proximity for large file transfer applications. The whole study is supported by real measurements carried out over Planetlab.**

I. Introduction

In Peer-to-Peer and overlay networks, the quality of service perceived by end-users can be optimized at the application level by identifying the best peer to contact or to take as neighbor. This requires to define a proximity function that evaluates how much two peers are close to each other from application point of view.

Different functions are introduced in the literature to characterize the proximity of peers, but most of them [1], [2], [3], [4] are based on simple metrics such as the delay, the number of hops or the geographical location. We believe that these metrics are not enough to characterize the proximity given the heterogeneity of the Internet in terms of path characteristics and access link speed, and the diversity of application requirements. Some applications (e.g., transfer of large files) are sensitive to other network parameters such as the bandwidth.

Therefore, the proximity should be defined at the application level taking into consideration the network metrics that decide on the application performance. To this end, we introduce the CHESS space where the proximity is characterized according to a utility function that models the quality perceived by peers at the application level. A peer is closer than another one to some third peer if it provides a better utility function, even if the path connecting it to the third peer is longer.

Based on extensive measurements over the Planetlab overlay network [5], we motivate our work by studying how much a proximity-based ranking of peers using the delay deviates from that using other network parameters (e.g., available bandwidth[2], loss rate[3]). Our observation is that the delay proximity is not always a good predictor of quality and that the other parameters have to be considered as well. Particulary, the best peer to contact is not always the closest one. Therefore, the knowledge of the different network parameters between peers helps in improving the performance of applications by allowing the definition of better proximity models.

The main constraint for determining the proximity in CHESS is how to infer the network parameters that impact the utility function in an easy and scalable way. In other terms, the estimation of the network parameters, between any two peers in a large system, should be achieved with a small measurement overhead and a limited cooperation among peers.

While there were several scalable models proposed recently for estimating the delay [3], [4], [6], [7], [8], [9], [10], there was to the best of our knowledge only one solution, called *BRoute*[13], for estimating the bandwidth. BRoute assumes that Internet bottlenecks are located on path edges (i.e., the first and the last 4 links of a complete IP level path) and are shared by many different paths. The tool proceeds first by measuring the available bandwidth over the edge links. Then, it estimates the end-to-end available bandwidth of a path as the minimum bandwidth of the link edges it includes. The identification of these edge links is done by calculating through BGP tables the shortest AS (Autonomous System) path between the two peers; then by mapping the edge links to this AS path. We believe that these tasks are challenging [14], [15], [16] and add a lot of complexity to the scheme.

In this work, we start by showing that the matrix factorization [32], [33] approach, initially proposed for estimating the end-to-end delay, is appropriate for estimating the end-to-end loss rate as well. However, this is not the case for the

[1]The notion of landmark in our context of bandwidth estimation is different from that used for delay estimation [3], [4], [8], [9]. Delay landmarks can be seen as reference points for inferring peer's network position. Bandwidth landmarks can be seen as intermediate nodes connecting peers with indirect paths that are suitable to the direct path bandwidth inference.

[2]Available bandwidth means the remaining bandwidth left on a path between any two peers. It is determined by the link having the smallest residual bandwidth. In the rest of the paper, we refer to this link by the term *tight* link. Also, we use the term bandwidth to refer to the end-to-end available bandwidth.

[3]We estimate the loss rate as the ratio of the number of lost packets and sent packets.

bandwidth which is far from being an additive metric. The bandwidth depends on the bottleneck link that may appear anywhere along an end-to-end path. Any model for bandwidth estimation has to identify the route connecting each couple of peers to be able to gauge the bottleneck link. The challenge is that such model must be scalable and easy to deploy.

Therefore, we propose a model for a scalable estimation of the bandwidth among peers that does not require AS-level path and edge link identification as in [13]. Our model estimates the bandwidth among peers using the bandwidth of the indirect paths that join them via a set of well defined proxies or relays that we call *landmark* nodes. Basically, the direct and the indirect path share the same tight link with some probability that depends on the location of the correspondent landmark with respect to the direct path or to one of the path end points. This probability is higher if the landmark is closer to one of the path end points. It can be also higher if the delay of the indirect path is nearer to that of the direct one. Thus, the bandwidth of each indirect path contributes to the estimation of the bandwidth of the direct one according to its assigned probability. In this way, one do not care if the bottleneck is on the edge links of the path joining the two peers or if it is in the middle. Moreover, a peer does not need to determine its AS-level source/sink trees to deduce the edge links that join it to the other peers as done in BRoute. Instead of that, peers have to identify the indirect paths that better represent the direct ones. This task is much easier to realize since we propose to obtain such information using the delay of the paths connecting peers to the landmarks.

Again, using Planetlab measurements, we evaluate the solution and analyze the impact of the location, and number of landmarks on the accuracy of the estimation. Our experiments show how a number of 40 to 50 landmarks is necessary for estimating the bandwidth among a worldwide distributed set of Planetlab nodes.

Finally, we compare the proximities in the delay and CHESS spaces from application point of view. A typical file transfer application is considered to evaluate the quality of service perceived by peers when they choose their neighbors based on these two distinguished proximity notions. We determine the proximity in CHESS among a worldwide distributed set of Planetlab nodes using a function that predicts the content transfer time. We observe that the characterization of this proximity, using our bandwidth estimation model, leads to a much better quality than that obtained when using the delay alone.

The outline of the paper is as follows. Next, we present our measurement setup. Section III motivates the need to consider the different network parameters when defining the proximity among peers. We review, in Section IV, the network embedding models proposed initially for estimating scalably the delay. We introduce, in Section V, a scalable model for estimating the bandwidth. In Section VI, we evaluate the performance gain achieved when considering the bandwidth estimations for determining the proximity in the CHESS space. The paper is concluded in Section VII.

II. Measurement setup

Our experiments consist of real measurements run over Planetlab platform [5]. Planetlab is an experimental network built between hundreds of academical sites (i.e., universities, research laboratories) worldwide distributed. Planetlab nodes have high bandwidth connectivity (i.e., on the order of $10/100$ Mbps). We do not claim that this platform is representative of all networks, but we believe that it is the best evaluative testbed available nowadays that satisfies the large scale feature that is required for our study. Moreover, this platform has proved its capability to be appropriate for measurements [17].

Our experiments on Planetlab consist in the following measurement sets. We take 127 Planetlab nodes spread over the Internet covering America, Europe, and Asia. Forward and reverse paths between each pair of Planetlab nodes are considered, which leads to 16002 measurements. For each unidirectional path between two Planetlab nodes, we measure the round-trip time D, the available bandwidth A, and the packet loss rate P. These measurements were repeated three times during the last week of February 2005. The measurement process is realized as follows.

Each measurement set consists of measuring the end-to-end network parameters of the paths connecting Planetlab nodes by performing direct probing among the nodes; which is done in a *Round-Robin* fashion. In other words, each Planetlab node i ($i = \{1, ..., 127\}$) probes the rest 126 Planetlab nodes consecutively. Once this is achieved, node $i+1$ takes its turn for probing the other nodes. Round-Robin probing has been applied in order to avoid parallel probing among nodes, which may have a large overhead on the network and subsequently that may change its characteristics. Next, we assume that the network characteristics remain steady during the Round-Robin measurements as if the probing action is realized in parallel. This is necessary for the validation of the models that will be described later along the paper.

The measurement is realized using the *Abing* tool [18]. This tool is based on packet pair dispersion technique [19]. It consists of sending a total number of 20 probe packet-pairs between the two end nodes of the measured path. Abing has the advantage of short measurement time on the order of the second. The short probing time is necessary for our experiments in order to achieve the Round-Robin measurement set in shorter interval of time[4]. Also, Abing tool provides a rich set of results (e.g, bandwidth in both directions), and a good functioning over Planetlab. The measurement accuracy provided by Abing on Planetlab is quite reasonable compared to other measurement tools [20]. All these features make the Abing tool appropriate for our study.

In the rest of the paper, we call a Planetlab node a peer. On the other hand, we select landmarks from a worldwide distributed set of Planetlab nodes in the following three ways:

- **Random**
  We randomly choose landmarks without considering their pairwise distances.

[4]In our settings, each Round-Robin measurement set takes 4 to 5 hours. But, network characteristics on the paths connecting Planetlab nodes does not frequently change [27]. Then, this may consolidate our assumption which consists of substituting the parallel measurements by the Round-Robin ones.

- **Maximum-distance**
  The maximum-distance algorithm consists in determining the landmark set whose summed pairwise distance is greedily maximized. This is realized as follows: (1) at the beginning, we randomly choose one peer as the first landmark, (2) from the rest of peers, we determine the next landmark as the one having the maximum average delay to the existing landmarks, (3) we repeat step (2) until reaching the required number of landmarks.
- **N-means**
  N-means is a well-known algorithm for grouping peers into N clusters and then choosing the closest peers to the clusters' centroids as landmarks. We identify the N-means landmark set with the following procedure: (1) initially, we randomly choose N peers as landmarks, (2) we assign each of the rest of peers to the cluster of the closest landmark, (3) we re-determine the N landmarks by identifying in each cluster the peer that minimizes the summed delay with the other peers in the same cluster, (4) we repeat step (2) and (3) until reaching an unchangeable set of landmarks.

III. Motivating CHESS

A. Overview

Different definitions were studied in the literature for characterizing the proximity among peers, and hence for selecting the appropriate peer to contact. These definitions can be classified into two main approaches static and dynamic. The difference between these approaches lies in the metric they consider. Static approaches [2], [21] use metrics that change rarely over time as the number of hops, the domain name and the geographical location. Dynamic approaches [1], [21], [4] are based on the measurement of variable network metrics. They mainly focus on the delay and consider it as a measure of closeness of peers; the appropriate peer to contact is often taken as the closest one in the delay space. The focus on the delay is for its low measurement cost (i.e., measurement time , amount of probing bytes). However, its use hides the implicit assumption that the path with the closest peer (in terms of delay) has the minimum (or relatively small) loss rate and the maximum (or relatively large) bandwidth.

While we believe that the delay can be an appropriate measure of proximity for some applications (e.g., non greedy delay sensitive applications or those seeking for geographical proximity), it is not clear if it is the right measure to consider for other applications whose quality is a function of diverse network parameters. Bandwidth greedy applications and multimedia ones are typical candidates for a more enhanced definition of proximity. To clarify this point, we use our measurements results and study the correlation among path characteristics. We want to check whether (i) the characteristics are correlated with each other, and (ii) how much a proximity-based ranking of peers using the delay deviates from that using other path characteristics.

As we will see in this section, there is a clear low correlation among path characteristics which motivates the need for an enhanced model for proximity. The closest peer in terms of delay is far from being optimal in the bandwidth space or loss rate space, and vice versa.

B. Delay vs. available bandwidth

Take a peer p and let p_d be the closest peer to p in terms of delay and p_a the best peer from p's point of view in terms of available bandwidth. First, we want to study how much the available bandwidth on the path connecting p to p_d, $A(p, p_d)$, deviates from the largest one measured on the path between p and p_a, $A(p, p_a)$. Figure 1(a) shows the CCDF (i.e., Complementary Cumulative Distribution Function) of the ratio $A(p, p_d)/A(p, p_a)$. The curve is calculated over all peers and for each data set. For a value x on the x-axis, the corresponding value on the y-axis gives the percentage of peers having on their path to the nearest peer an available bandwidth larger than x times the maximum available bandwidth.

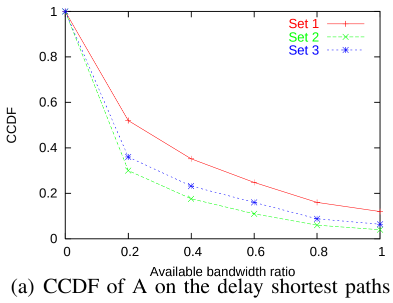
We can see that a small percentage of peers, between 5% and 15%, have the maximum bandwidth on their path with the nearest peer. A large proportion of peers, around 80%, have less than 50% of the maximum bandwidth on this path. This indicates that selecting the best peer in terms of delay leads in most cases to an available bandwidth far from the optimal. Applications having a high bandwidth requirement could suffer from this choice.

In our setting, the delay and bandwidth are lightly negatively correlated[5] with a coefficient equal to -0.01. This can be also observed in Figure 1(b) where we plot the available bandwidth for peers of rank r in the delay space, r varying from 1 to 126, averaged over the 127 peers. The figure is plotted for the three data sets. We note that looking at farther and farther peers in the delay space does not lead to an important decrease in the available bandwidth, and so there is a high chance of having the optimal peer from bandwidth point of view located far away (in the delay space) from the peer requesting the service.

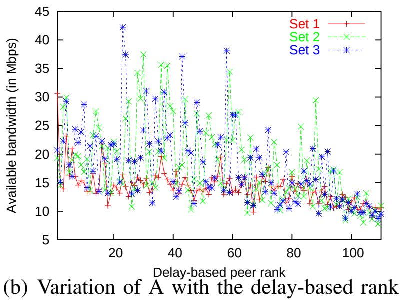
C. Delay vs. loss rate

Applications are sensitive to the loss rate. We want to check in this section how well a definition of proximity based on delay satisfies the loss rate. We find that all peers have, in our setting, a null loss rate ($P = 0$) on their path with at least one other peer. To check whether the nearest peer results in the minimum loss rate (i.e., zero), we plot in Figure 2(a) the CDF of the loss rate on the path connecting a peer p to its nearest peer p_d. The distribution is computed over the 127 peers and for each data set. We can see that around 89% of peers have the minimum loss rate ($P = 0$) on their path to the nearest peer. Our measurements show that the delay and loss rate are positively correlated with a coefficient equal to 0.38. Another observation we made is that long as we move away from a peer in the delay space, the loss rate jumps to values on the order of several percents, then it increases slowly. This is well illustrated in Figure 2(b) where we plot the packet loss rate on the path connecting a peer to its neighbor of rank r in the

[5]One would have expected a coefficient closer to -1 than to 0.

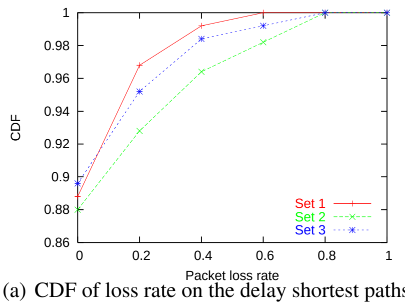


(a) CCDF of A on the delay shortest paths

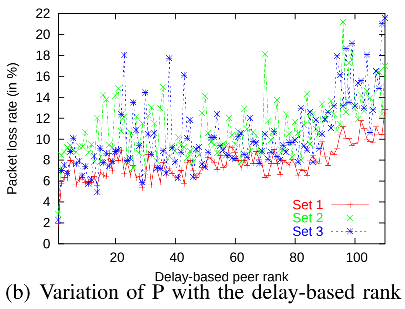


(b) Variation of A with the delay-based rank

Fig. 1. Delay versus available bandwidth



(a) CDF of loss rate on the delay shortest paths



(b) Variation of P with the delay-based rank

Fig. 2. Delay versus loss rate

delay space, r changing from 1 to 126. The figure is averaged over the 127 peers and plotted for the three data sets.

In summary, the closest peer seems to give the minimum loss rate in most cases. The reason could be the fact that both are located in a non-congested neighborhood. Now, when it comes to selecting more than one peer for a certain service sensitive to the loss rate, taking the delay as a metric of proximity stops being efficient, and the loss rate has to be considered as well.

D. Discussion

The weak correlation among path characteristics pointed out by our measurements motivates us to introduce the proximity in CHESS. It consists in determining the proximity at the application-level by estimating some utility function that models the application quality such as the transfer time for file transfer applications. Peers are ranked with respect to each other using the values given by the utility function for the paths joining them. In [31], we introduced CHESS for the first time and validate it using direct path measurements. Current work is about how to estimate the network parameters included in the utility function in an easy and scalable way.

IV. Network embedding

A. Overview

Inferring network parameters (e.g., delay, available bandwidth, loss rate) among a large number of peers without achieving a mesh-based measurements is a hard problem. Many approaches [3], [4], [6], [7], [8], [9], [10] have been proposed for estimating the end-to-end delay among peers from a set of partially observed measurements. Most of these solutions are based on the network embedding. It consists in assigning to each peer a position (i.e., a vector of coordinates) in a low dimensional space where the delay between any two peers is estimated by their Euclidean distance. Peers' coordinates are usually deduced from delay measurements to a number N of landmarks $L\{L_1, ..., L_N\}$.

More formally, suppose that the network contains n peers $p = \{p_1, p_2, ..., p_n\}$. The delay on the paths joining peers is represented by an $n \times n$ matrix D, where D_{ij} is the measured delay from p_i to p_j. A network embedding is a mapping $C : p - \mathbb{R}^d$ in the following way:

$$D_{ij} \approx \|\overrightarrow{C}_i - \overrightarrow{C}_j\|, \forall i, j = 1, ..., n \quad (1)$$

where $\overrightarrow{C}_i = (C_{i1}, C_{i2}, ..., C_{id})$ is the coordinate vector of p_i giving its position in a d-dimensional Euclidean space. Thus, the delay between any two peers p_i and p_j is estimated as the Euclidean distance between their coordinates:

$$\widehat{D}_{ij} = \|\overrightarrow{C}_i - \overrightarrow{C}_j\| = \left(\sum_{k=1}^{d}(C_{ik} - C_{jk})^2\right)^{\frac{1}{2}} \quad (2)$$

The challenge of network embedding is to assign a coordinate vector C_i to each peer p_i from a partially observed delay matrix D. Global Network Positioning System (GNP) [3] was the first work in this field. In GNP, peers measure the delay to a set of well-known landmark nodes. Then, the Simplex Downhill method [25] is used to deduce peers' coordinates by minimizing an objective function representing the error between the estimated and the measured delays. The same algorithm has been applied in PIC [26], after investigating issues related to security. In [1], Vivaldi simulates the overlay network by a network of physical springs. It proposes to determine peers' coordinates in a distributed way without the need to dispose of a fixed set of landmarks. Each peer contacts a random set of peers and adjusts its coordinates permanently until minimizing the potential energy of a spring system. ICS [10] and Virtual Landmarks [4] are based on Lipschitz embedding and Principle Component Analysis (PCA). They embed the peers in a low dimensional Euclidean space characterized using PCA.

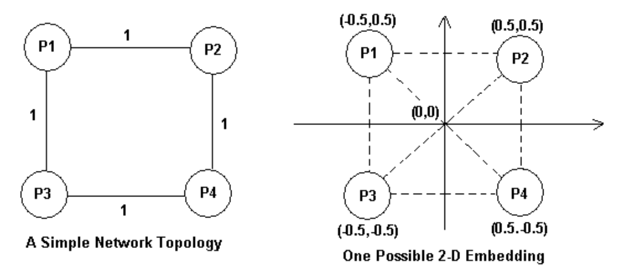


Fig. 3. A network embedding example

These coordinates-based mechanisms suffer particularly from the fact that they provide Euclidean distances which are symmetric and satisfy the triangle inequality. This may not be consistent with the real network topology [27], [28], [29], [30]. Moreover, the authors in [32] indicate one more limitation that some peers probably do not have a direct path joining them. In this case, the estimated delay of these paths is inaccurate. This is illustrated in the example of Figure 3 where we present a simple network topology containing four peers connected only to their neighbors by unit delays. The estimated delays in the two-dimensional embedding are $\widehat{D}_{14} = \widehat{D}_{23} = \sqrt{2}$ while the real delays in this topology are $D_{14} = D_{23} = 2$. Similar cases arise in tree-based topologies. In such cases, it may that there is no Euclidean embedding that can model exactly the real network delays.

B. Distance Matrix Factorization

In [32], [33], the authors propose to infer the network delay in a way that is able to model the sub-optimal and asymmetric routing that may exist in practice. This model is based on the distance matrix factorization. Next, we describe briefly the model and then we study its capacity to estimate the delay, the loss rate and the available bandwidth.

Basically, nearby peers are expected to have similar delay distances to all the other peers. Then, an $n \times n$ delay matrix may contain dependent rows[6]. From linear algebra, such matrix can be expressed as the product of two smaller matrices:

$$D \approx XY^T, \tag{3}$$

where X and Y are $n \times d$ matrices with $d \ll n$. In contrast with the coordinate-based approaches that assign to each peer a coordinate vector, a delay matrix factorization associates to each peer p_i two vectors $\overrightarrow{X_i}$ and $\overrightarrow{Y_i}$ of d dimensions. $\overrightarrow{X_i}$ is called the outgoing vector and $\overrightarrow{Y_i}$ the incoming vector for peer p_i. Hence, the estimated delay from any peer p_i to any peer p_j is simply the dot product between the outgoing vector of p_i and the incoming vector of p_j.

The factorization of D $(n \times n)$ into two matrices of smaller dimension can be done by its singular value decomposition (SVD). The singular value decomposition of D can be expressed as:

$$D = USV^T, \tag{4}$$

[6]Dependent rows are equal rows or rows that can be expressed as a linear combination of other rows.

where U and V are $n \times n$ orthogonal matrices, and S is an $n \times n$ diagonal matrix. Calculating SVD consists in finding the eigen values and eigen vectors of the matrices DD^T and D^TD. The matrix S contains the singular values of D ranked in a decreasing order[7] . The columns of the matrices U and V are respectively the eigen vectors of DD^T and D^TD.

When the number of dependent rows in D increases, the number of principle components decreases and subsequently the number of singular values that are significant in magnitude decreases as well. This is due to the property of the singular values that measure the significance of the contribution from each principle component. After eliminating the null and negligible values, the remaining singular values get a number d which is less or equal to n.

Thus, an $n \times n$ delay matrix D can be decomposed into two smaller matrices X $(n \times d)$ and Y $(n \times d)$ which are computed as the following:

$$X_{ij} = U_{ij}\sqrt{Sjj}, \tag{5}$$

$$Y_{ij} = V_{ij}\sqrt{Sjj}, \tag{6}$$

where $i = 1...n$ and $j = 1...d$. This leads to the matrix $\hat{D} = XY^T$ which is a low-rank approximation to the real delay matrix D. This approximation depends obviously on the choice of d which can be determined by minimizing the following squared error function:

$$\sum_i^n \sum_j^n (D_{ij} - \overrightarrow{X_i} \cdot \overrightarrow{Y_j})^2, \tag{7}$$

where $\overrightarrow{X}_i \in \mathbb{R}^d$ and $\overrightarrow{Y}_j \in \mathbb{R}^d$.

1) Distance reconstruction: We evaluate the error occurred from such approximation for different values of the reduced dimension d. This is done for the delay, loss rate, and bandwidth matrices. Note that in the literature only the delay was studied. We want to check whether the matrices for the other metrics can be compressed as well. This is important for the design of the scalable CHESS space. We construct the three matrices using our three measurement sets carried out over the 127 Planetlab nodes. The diagonal of each constructed matrix contains null values. These null values make difficult the matrix dimensionality reduction which depends on the linear dependency among rows. Therefore, we interchange the diagonal values by values that simulate a peer clustering process. For the delay and loss rate matrices, we fill the diagonal values (resp. D_{ii}, and P_{ii}) by the minimum values of the correspondent rows (resp. $min_{j=\{1..127\}}D_{ij}$, and $min_{j=\{1..127\}}P_{ij}$). For the available bandwidth matrices, we fill the diagonal values (A_{ii}) by the maximum value of the correspondent rows ($max_{j=\{1..127\}}A_{ij}$).

We plot in Figure 4, the mean reconstruction error of these matrices as a function of the reduced dimension d. For different values of the reduced dimension d on the x-axis, the y-axis draws the mean relative error (MRE) of the approximated matrix values (estimated values $\hat{D}_{ij}, i, j = \{1...n\}$) with respect to the real values (i.e., measured values

[7]The singular values are calculated as square roots of the nonzero eigen values of DD^T.

$D_{ij}, i, j = \{1...n\}$). More formally, MRE is computed as the following:

$$MRE = \frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} \frac{|\hat{D}_{ij} - D_{ij}|}{D_{ij}} \tag{8}$$

Figure 4 shows how the reconstruction error (i.e., mean relative error MRE on the y-axis) decreases when the matrix dimension is less and less compressed (i.e., the value of d on the x-axis increases). For the delay and loss rate matrices, the figure shows that the reconstruction error is small even for a large dimensionality reduction (as for $d = 10$, MRE between 0.4 and 0.6). This is not the case for the available bandwidth matrix where the reconstruction error is more than twice that obtained for reconstructing the delay and loss matrices. This can be explained by the fact that the delay and loss parameters are additive and subsequently it is expected to find a linear dependency among their vectors. This is not the case for the bandwidth parameter.

2) Distance prediction: We apply the matrix factorization technique for estimating the network parameters among peers that have not been used for constructing the network matrices. This is already done for the delay, in [32], where the authors estimate the delay on the path between two peers p_i and p_j as the dot product between the delay outgoing vector of p_i and the delay incoming vector of p_j. The two vectors are calculated separately by probing a set of landmarks. In that paper, the authors show that their matrix factorization model provides more accurate delay estimations comparing to the other network embedding approaches. We wonder whether this model is able to estimate accurately the loss rate and the bandwidth as well. Therefore, we take 40 nodes, out of the 127 Planetlab nodes, as landmarks[8] and the rest 87 nodes as peers.

We begin by constructing the delay and bandwidth matrices using the measurements achieved among the 40 landmarks over our three datasets. The landmarks are selected in three ways: randomly, according to the maximum-distance algorithm, and according to the N-means algorithm. More details concerning these algorithms are described in Section II.

Then, we determine the delay, loss, and bandwidth outgoing and incoming vectors for each peer from the measurement it conducted to the landmarks. First, landmarks compute their vectors by measuring the paths among each other (we take $d = n = 40$). Next, peers infer their vectors in a distributed way. We explain how this to be done for the delay. The loss and bandwidth vectors are calculated similarly.

For a given landmark set, each peer p has to measure the delay to and from each landmark. The delay from p to landmark L_i is denoted by D_i^{out} and that from the landmark L_i to p by D_i^{in}. The outgoing ($\overrightarrow{X}_{new}$) and incoming ($\overrightarrow{Y}_{new}$) vectors of p should satisfy the following equations:

$$D_i^{out} = \overrightarrow{X}_{new} \cdot \overrightarrow{Y}_i \tag{9}$$

$$D_i^{in} = \overrightarrow{X}_i \cdot \overrightarrow{Y}_{new} \tag{10}$$

[8]Previous studies show that 20 landmarks can be enough for estimating the delay among a wide distributed set of peers.

The solution of these equations can be obtained by minimizing the square error summed over all landmarks:

$$\overrightarrow{X}_{new} = arg_{\overrightarrow{U} \in \mathbb{R}^d} Min \sum_{i=1}^{n} (D_i^{out} - \overrightarrow{U} \cdot \overrightarrow{Y}_i)^2 \tag{11}$$

$$\overrightarrow{Y}_{new} = arg_{\overrightarrow{U} \in \mathbb{R}^d} Min \sum_{i=1}^{n} (D_i^{in} - \overrightarrow{X}_i \cdot \overrightarrow{U})^2 \tag{12}$$

After determining the delay (resp. loss and bandwidth) outgoing and incoming vectors, we estimate the delay (resp. loss and bandwidth) between each pair of the 87 peers by the scalar dot product of their vectors. We evaluate the mean and the standard deviation of the absolute relative error (denoted respectively by MRE and $stdRE$) of the estimated delay (resp. estimated loss and bandwidth) compared to the measured values. Tables IV-B.2, I, II show MRE and $stdRE$ of the delay, loss and bandwidth estimation error obtained when using the different landmark sets and over our three datasets. The table shows that the estimation accuracy is quasi-similar when landmark sets are chosen using N-means, and max-distance algorithms. In these cases, MRE and stdRE for the delay estimation varies between 0.35 and 0.5. With the random landmark set, the estimations are less accurate. We note that these results are coherent with those obtained in [32]. Besides, we observe in the tables a good estimation accuracy for the loss rate parameter when using the N-means and max-distance landmark sets; MRE and $stdRE$ varies respectively in the intervals $[0.25, 0.5]$ and $[0.3, 0.7]$. For the bandwidth, the table shows that the estimations are not as accurate as the delay and loss ones. A relatively large estimation errors are obtained for the different landmark and measurement sets; MRE and stdRE vary respectively in the intervals $[1.5, 2.5]$ and $[1.5, 5]$.

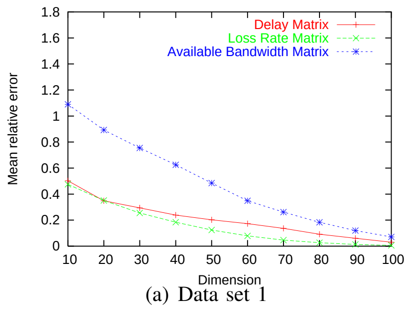
Thus, while the matrix factorization approach provides accurate estimations for the delay and loss rate parameters, it is not the case for the bandwidth. This is because the bandwidth is not an additive metric and it rather depends on the bottleneck link that may appear anywhere along an end-to-end path. This means that network embedding approaches are not expected to be the appropriate tools for estimating this parameter. One may conclude that a bandwidth estimation model must identify the route connecting each couple of peers to detect its bottleneck link. The challenge is that such model must be scalable and easy to deploy.

V. Scalable end-to-end bandwidth inference

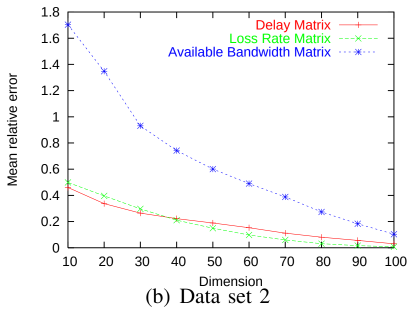
A. Model Overview

Our model[9] consists in inferring the bandwidth between any pair of peers based only on their bandwidth vectors. A peer obtains its bandwidth vector by measuring the direct and reverse bandwidth on its path to the landmarks. Hence, the scheme is scalable since its overhead is linear with the number of peers in the system. Also, it is easy to implement since (i) peers do not need to know and probe each other; any node can

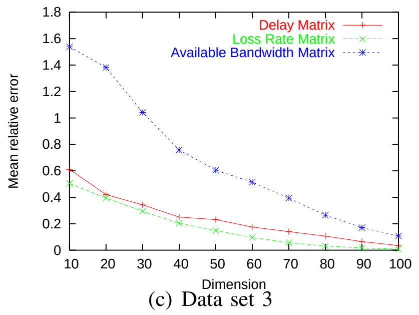
[9]We have introduced our bandwidth estimation model previously in [11], [12].



(a) Data set 1



(b) Data set 2



(c) Data set 3

Fig. 4. Matrix reconstruction error after the reduction to the different dimensions over the 3 Planetlab datasets

MRE	Random	Max-distance	N-means
Delay	1.2	0.52	0.5
loss	0.9	0.355	0.24
ABw	1.4	1.85	2.8
stdRE	Random	Max-distance	N-means
Delay	0.66	0.4	0.43
loss	0.8	0.728	0.43
ABw	2.7	1.6	2.7

TABLE I
ESTIMATION ERROR BY SVD OVER DATASET 1

MRE	Random	Max-distance	N-means
Delay	1.19	0.49	0.47
loss	0.65	0.28	0.25
ABw	2.5	2.3	2
stdRE	Random	Max-distance	N-means
Delay	0.71	0.4	0.41
loss	0.43	0.32	0.34
ABw	4.8	5	4.7

TABLE II
ESTIMATION ERROR BY SVD OVER DATASET 2

MRE	Random	Max-distance	N-means
Delay	1.3	0.49	0.5
loss	0.54	0.275	0.244
ABw	1.9	1.6	1.8
stdRE	Random	Max-distance	N-means
Delay	0.70	0.37	0.42
loss	0.4	0.43	0.42
ABw	3.6	4.2	3.9

TABLE III
ESTIMATION ERROR BY SVD OVER DATASET 3

estimate the bandwidth between any two peers based on their bandwidth vectors, (ii) no need for the routing information used in [13] (described in Section I).

For a couple of peers, we denote by (i) *direct path* the network path that joins them directly using IP routing, and by (ii) *indirect path* the path that joins them by passing by a landmark node. We note that N indirect paths (i.e., N being the number of landmarks) are assigned to each direct path.

We estimate the end-to-end bandwidth of a path joining two peers using the following class of linear functions:

$$EB = \sum_{i=1}^{N} w_i \cdot BI_i, \tag{13}$$

where BI_i is the bandwidth of the indirect path that passes by the landmark L_i, and w_i is the normalized weight (i.e., $\sum_{i=1}^{N} w_i = 1$) assigned to this indirect path according to the location of its corresponding landmark with respect to the two peers. The idea behind this definition of the estimation function is as follows. We consider that the direct path shares the same tight link with the indirect path that passes by the landmark L_i with some probability w_i. This probability depends on the location of the corresponding landmark with respect to the direct path or to one of its end points. w_i is higher if for example L_i is closer to one of the path end points (to be validated in the next section). By varying the expression of the probability w_i, we are able to cover different policies for bandwidth estimation ranging from the one that gives the same priority to all landmarks to the one that privileges the landmark that we deem the most suitable for bandwidth inference.

For example, take the case of Figure 5, where peers $\{p_1, p_2, p_3, p_4\}$ are connected to the *network core* via a set of path edges. We suppose that there are three landmarks distributed in the network core as intermediates nodes for bandwidth inference. Thus, for the direct path joining p_1 to p_3 (p_1p_3), there are three associated indirect paths $\{1, 2, 3\}$. Each indirect path i (i.e., i = {1,2,3}) is composed of the two IP path portions p_1L_i and L_ip_3.

Suppose that the direct path p_1p_3 passes by the routers $\{R_{1a}, R_{1b}, R_{1e}, R_{1i}, R_{3i}, R_{3d}, R_{3b}, R_{3a}\}$ and by some set of routers in the network core. In this case, the direct path p_1p_3 overlaps with (i) the indirect path 1 by at least the two path portions $\{p_1R_{1a}R_{1b}R_{1e}R_{1i}\}$ and $\{R_{3i}R_{3d}R_{3b}R_{3a}p_3\}$, (ii) the indirect path 2 by at least the two path portions $\{p_1R_{1a}R_{1b}\}$ and $\{R_{3b}R_{3a}p_3\}$, (iii) the indirect path 3 by at least the two path portions $\{p_1, R_{1a}\}$ and $\{R_{3a}, p_3\}$. Hence, the bandwidth of the indirect path 1 must be assigned the largest weight when estimating the bandwidth of the direct path p_1p_3 since these two paths have the most common portions and subsequently it is more probable that they share the same bottleneck link.

In our description, we focus on path edges since it is very probable that the bottleneck is located on these edges as shown in [13]; the authors of [13] obtain that it is almost 4 hops away from the path end points. But contrary to [13], our solution applies to other contexts where the bottleneck is not solely on the edge. The key point is that estimating the bandwidth on the path between two peers depends mainly on the location of the landmarks with respect to these peers. This dependency is explored in the next section.

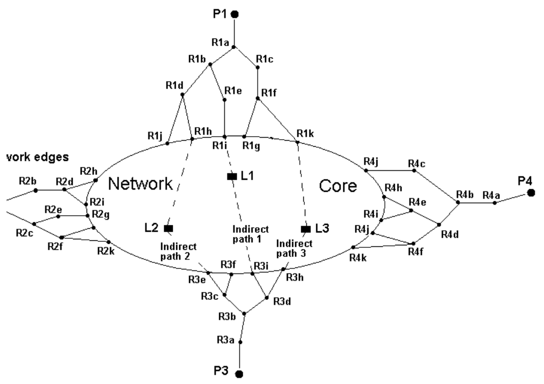


Fig. 5. Direct versus Indirect paths

B. Impact of landmarks' locations

To evaluate the impact of landmarks' locations on the bandwidth estimation accuracy, we consider the following real scenario conducted over Planetlab. We take 8 Planetlab nodes distributed in different European countries as landmarks. We also take 14 Planetlab nodes completely distributed in Europe as peers. We ask the question of whether the European landmarks can help to estimate accurately the bandwidth on the path between European peer and any other peer. Therefore, each of the European peers measures the RTT and the direct and reverse available bandwidth to 34 Planetlab nodes distributed worldwide. This leads to 476 measured paths. Then, we infer the bandwidth of these paths using Equation (13) and we compare the estimations with the measured values. This comparison is done for different weights in the estimation function (Equation 13).

Our landmark nodes are chosen with the main concern to have a high bandwidth connectivity to the Internet. This is an important requirement since we want to avoid having the bottleneck, of an indirect path, decided by the landmark itself.

We consider different forms of the probability w_i, and subsequently of the end-to-end bandwidth estimation function. By doing that, we are able to study the correlation between the estimation accuracy and the locations of the landmarks. We divide the study into two main parts: (i) the estimation function depends on the delay closeness between the direct path and the indirect paths, (ii) the estimation function depends on the delay closeness between the landmarks and the path end points.

1) Estimating bandwidth based on indirect paths' delays: One possibility is to estimate the end-to-end bandwidth of a direct path using that of the indirect path having the shortest delay. Even though we found satisfactory results, we believe this method is not sufficient for providing accurate estimation since direct IP routing may lead to an end-to-end delay larger than the one of the shortest indirect path, with both paths having different sets of links and hence different bottlenecks. The accuracy could improve by considering more than one indirect path in the estimation function while assigning more weight to those having shorter delays. This consideration is mainly recommended when there are more than one indirect path having delays on the order of that of the shortest one.

Thus, we consider all the N indirect paths in the bandwidth estimation function (Equation 13) with the following expression for the weight w_i:

$$w_i = \frac{C_i}{\sum_{i=1}^{N} C_i}, \quad \text{for } i = \{1, .., N\} \tag{14}$$

where,

$$C_i = \left(\frac{RI_{min}}{RI_i}\right)^{\alpha}, \tag{15}$$

RI_i is the round trip delay of the indirect path that passes by the landmark L_i, RI_{min} is that of the shortest indirect path among the N indirect paths, and α is a positive real number.

Hence, we obtain the following first expression of the estimation function:

$$EB_1 = \sum_{i=1}^{N} \frac{\left(\frac{RI_{min}}{RI_i}\right)^{\alpha}}{\sum_{i=1}^{N}\left(\frac{RI_{min}}{RI_i}\right)^{\alpha}} \cdot BI_i \tag{16}$$

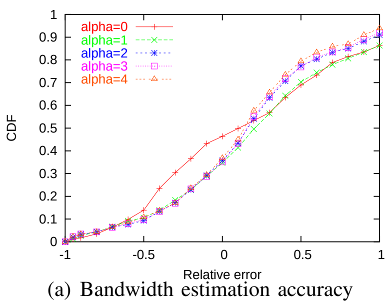
We draw in Figure 6(a) the CDF of the relative bandwidth estimation error which is calculated as:

$$\frac{EB_1 - A_{measured}}{A_{measured}}, \tag{17}$$

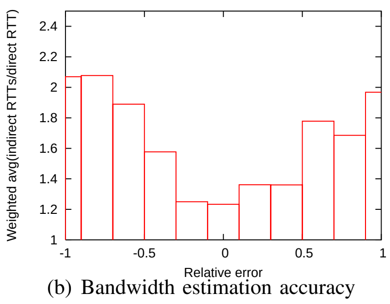
for all the bandwidth estimations and for different values of α. The figure shows that when the α parameter increases, the estimation accuracy improves. This is expected since when $\alpha = 0$, the bandwidth component of all the indirect paths gets the same probability, and when α becomes large, the indirect paths having shorter delays, and hence better representation of the direct path, get more probability than those having larger delays. For $\alpha > 3$, we observe that the results become steady. This can be explained by the fact that the estimation becomes only dependent on the indirect paths having a delay on the order of that of the shortest indirect path. For $\alpha = 4$, the figure shows that approximatively 40% of the estimations are accurate within 25% and 70% of the estimations are accurate within 50%.

To show the correlation between the estimation accuracy and the difference in the delay between the direct and the indirect paths, we plot Figure 6(b) for the case $\alpha = 4$. For an estimation error interval (on the x axis) of length 0.2, the y axis shows the sum $\sum_{i=1}^{N}(w_i \cdot RI_i)/R_d$, which is a weighted average of the ratio of the indirect paths' delays and the direct path delay (R_d). This sum is averaged over each interval of length 0.2. The figure shows a clear correlation between the two entities plotted on the x and y axis. This means that when some landmarks are located such that the delay of their corresponding indirect paths is close to that of the direct path, the estimation accuracy is high.

2) Estimating bandwidth based on the delay distance between landmarks and peers: Now, instead of relying our estimation on the end-to-end delay of the indirect paths, we focus on how close landmarks are to the direct path end points. Thus, for each pair of peers, we consider the N indirect paths in the bandwidth estimation model after assigning more weight for those going through landmarks that are closer to any of the two peers. Basically, we want to check if these latter indirect paths are more representative of the direct path than the ones

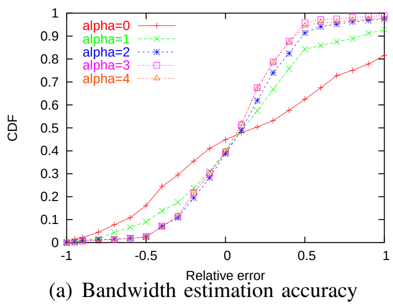


(a) Bandwidth estimation accuracy

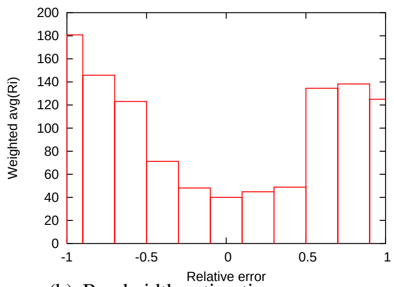


(b) Bandwidth estimation accuracy

Fig. 6. Bandwidth estimations based on indirect paths' delays



(a) Bandwidth estimation accuracy



(b) Bandwidth estimation accuracy

Fig. 7. Bandwidth estimations based on the delay distance between landmarks and peers

having smaller end-to-end delays. We express the coefficients C_i as:

$$C_i = \left(\frac{R_{min}}{R_i}\right)^{\alpha}, \tag{18}$$

where,

$$R_i = min(R_{xi}, R_{yi}), \tag{19}$$

R_{xi} represents the round trip delay between the peer x and the landmark L_i, and

$$R_{min} = min_{i=1..N} R_i. \tag{20}$$

We recalculate the w_i function (Equation 14) and subsequently the estimation function (Equation 13) with these new coefficients C_i. Thus, we obtain the following second expression of the estimation function:

$$EB_2 = \sum_{i=1}^{N} \frac{\left(\frac{R_{min}}{R_i}\right)^{\alpha}}{\sum_{i=1}^{N}\left(\frac{R_{min}}{R_i}\right)^{\alpha}} \cdot BI_i \tag{21}$$

Then, we plot in Figure 7(a) the CDF of the relative estimation error which is calculated as:

$$\frac{EB_2 - A_{measured}}{A_{measured}}, \tag{22}$$

for all the bandwidth estimations and for different values of α. As before, when α increases, the indirect paths having landmarks close to one of the two peers get more weight. The results shown in the figure become stationary for $\alpha > 3$. This is because the bandwidth estimations become only dependent on the few indirect paths having landmarks close to one of the peers. The figure shows better results comparing to the previous cases studied; around 57% of the estimations are accurate within 25% and 93% of the estimations are accurate within 50%.

To show the correlation between the estimation accuracy and the landmarks' closeness to the extremities, we plot Figure 7(b) for the case $\alpha = 4$. For an estimation error interval (on the x axis) of length 0.2, the y axis shows $\sum_{i=1}^{N} w_i \cdot R_i$ averaged over the estimations inside the interval. The figure shows a clear correlation between the two entities in the x and y axis. This means that when some landmarks (among the N) are close to the path extremities, the estimation accuracy improves. Furthermore, it becomes better than the case where the estimation depends on the shortest indirect paths (see Figures 6(a) and 7(a)).

This is due to the fact that the indirect paths, going through landmarks which are close to path extremities, are more expected to provide better representation of the direct path since it is more probable that IP will route through them. In such cases, we recommend the use of the estimation function EB_2 presented in this section. But for the couples of peers that are relatively far from all the landmarks, the delay closeness between landmarks and peers is not expected to be anymore a good criterion for testing the representation of the indirect paths with respect to the direct path. In such situations, the location of the landmarks with respect to the direct path can be more helpful for this purpose. Therefore, we propose the use of EB_1 (described in Section V-B.1) which depends on the shortest indirect paths and it is expected to estimate accurately the bandwidth when the delays of some indirect paths are close to that of the direct one (as obtained in Figure 6(b)); even if landmarks are far from path end points.

C. *Bandwidth estimation function*

Hence, for each couple of peers, we apply the following statement for determining the expression of the bandwidth estimation function (EB):

$$EB = \begin{cases} EB_2 & \text{if } R_{min} < R_{threshold}, \\ \\ EB_1 & \text{elsewhere}, \end{cases} \quad (23)$$

where R_{min} is expressed in Equation 20, $R_{threshold}$ is a threshold delay for examining the closeness between landmark and peers, EB_1 and EB_2 are expressed respectively in Equations (16) and (21).

Thus, to estimate the bandwidth on the path between two peers, we first check the delay closeness between landmarks and peers by the statement $R_{min} < R_{threshold}$. We take $R_{threshold} = 80ms$ based on what we have obtained in Figure 7(b). In that figure, the estimations, that are within the 50% accuracy, belong to the cases where at least one landmark is close to one of the path endpoints by a distance which is less than $80ms$ (i.e., $R_{min} < 80ms$). In such cases, the estimation function EB_2, which depends on the delay distance between landmarks and peers, is preferable to be considered for inferring the bandwidth. Otherwise, landmarks are relatively far from the path extremities and subsequently it becomes more helpful to rely on the location of the landmarks with respect to the direct path. In such cases, we use the estimation function EB_1 where the delay closeness between the direct and indirect paths is the criterion.

The challenge appears when the overlay network contains a large number of peers widely distributed. In this case, to infer accurately the bandwidth on the path between any pair of peers, a larger number of landmarks is obviously required for our estimation model. This issue is explored in the next section.

D. Impact of the number of landmarks

Instead of 8 landmarks distributed in Europe, we now infer the available bandwidth, on the paths joining a worldwide set of peers, using different landmark sets having the numbers $N = \{10, 20, 30, 40, 50\}$ distributed worldwide. Each landmark set is selected from 100 nodes having the highest bandwidth connectivity among the 127 Planetlab nodes[10]. Landmark selection is realized randomly, using the max-distance algorithm, and using the N-means algorithm. See Section IV-B for more details concerning these algorithms.

We plot in Figure 8 and 9 the mean (MRE) and the standard deviation ($stdRE$) of the bandwidth estimation error for the different landmark sets and over our three datasets. MRE is calculated as follows. For each set of N landmarks chosen from the 127 peers as described before, we infer the bandwidth on the paths joining the rest of peers of number $127-N$. This is done using the bandwidth estimation function described in Section V-C. Then, MRE (resp. $stdRE$) is computed as the mean (resp. the standard deviation) of the absolute relative errors of the estimated values with respect to the measured values (as computed in 8).

Figures 8 and 9 show that the bandwidth estimation error decreases when the number of landmarks increases. This is expected since with a wider coverage of the landmarks it becomes more probable to find indirect paths which better represent the direct ones. Besides, the smaller bandwidth estimation error is obtained when using the N-means landmark sets. MRE (resp. $stdRE$) is between 1.3 and 1.6 (resp. between 3 and 4) for N equal 10 N-means landmarks and it decreases when N increases to become between 0.2 and 0.5 (resp. between 0.2 and 0.4) for N equal 50.

[10]We remind that this is an important requirement since we want to avoid having the bottleneck, of an indirect path, decided by the landmark itself.

The estimations obtained when using the max-distance sets of landmarks are not as accurate as the case of the N-means sets of landmarks. In the former case, MRE (resp. $stdRE$) is between 1.6 and 2.1 (resp. between 5 and 7.1) for N equal 10 max-distance landmarks and it decreases when N increases to become between 0.5 and 0.9 (resp. between 0.4 and 1) for N equal 50. The worst case is obtained when using the random sets of landmarks. In this last case, MRE (resp. $stdRE$) is between 1.2 and 1.4 (resp. between 2.7 and 3.2) for N equal 50.

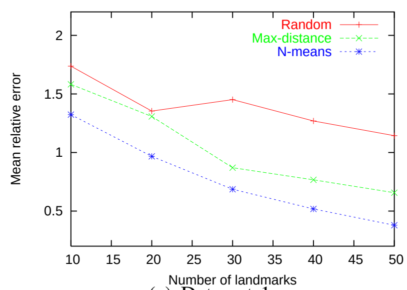
Moreover, we observe in the figures that it may happen that for some number of random landmarks, MRE and $stdRE$ are larger than those obtained when a lower number of random landmarks is used. This can be caused by the fact that the smaller random set has a wider coverage than the larger random one. One may conclude that the distribution of the landmarks is as important as their number. A large number of randomly chosen set of landmarks may not be appropriate for bandwidth inference if they have a distribution covering a small portion of the network space while the peers cover all the space. In this case, a large number of direct and indirect paths will be disjoint and subsequently a large number of inaccurate estimations may occur. We note that when the indirect paths do not overlap with the direct one, the model provides accurate estimation only if the path bottleneck holds on the network connectivity of the peers. Hence, a random set of landmarks could be inappropriate for the bandwidth estimation model even if it contains a large number of nodes.

This is not the case when using the N-means landmark sets. N-means consists in grouping the peers into N clusters and then choosing the closest peers to the clusters' centroids as landmarks. Such delay minimization between landmarks and peers leads to better estimate the bandwidth on the paths joining peers as obtained in Section V-B.2.

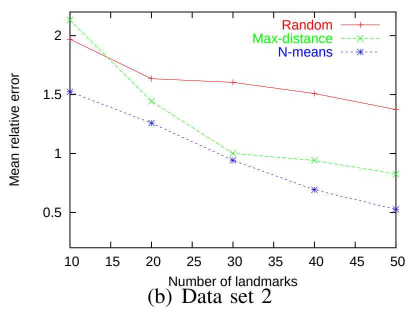
Thus, in our settings, a set of 40 to 50 landmarks having an N-means distribution is appropriate for estimating the bandwidth among a worldwide distributed set of Planetlab nodes. In the next section, we evaluate the performance gain achieved when considering the proximity in CHESS, determined using our bandwidth estimation model, instead of the delay proximity.

VI. Enhanced Proximity perceived by the application: A case study

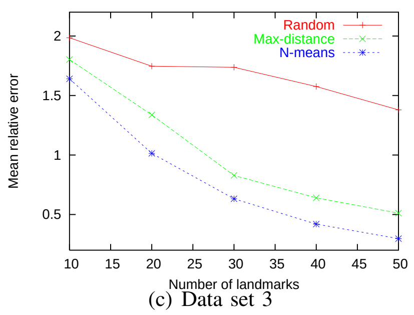
From the standpoint of a certain peer, we recall that peers are ranked within the CHESS space in a decreasing order of the utility function. Thus, close peers in the CHESS space are those providing the best application quality despiting their network locality. In[31], we have obtained that the proximity in CHESS outperforms the basic delay proximity for different



(a) Data set 1

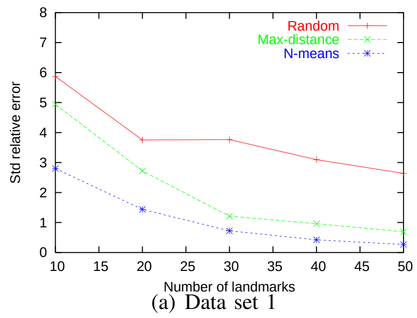


(b) Data set 2

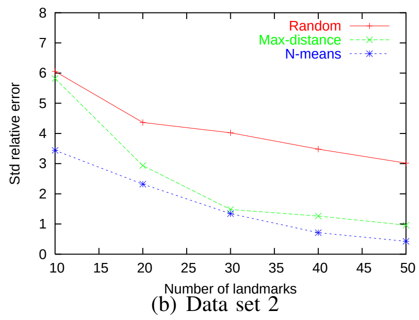


(c) Data set 3

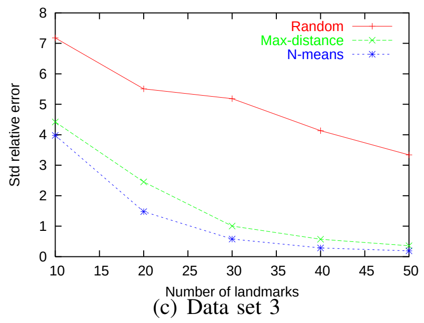
Fig. 8. Mean estimation error obtained for different number of landmarks chosen in a random, max-distance, and N-means ways



(a) Data set 1



(b) Data set 2



(c) Data set 3

Fig. 9. Standard deviation of the estimation error obtained for different number of landmarks chosen in a random, max-distance, and N-means ways

applications. This is obtained using the measured values of the network parameters. However, determining the proximity of peers using direct probing among them is not efficient due to its heavy overhead in large scale networks. Therefore, we use in this section our scalable bandwidth estimation model (described in Section V-A) for characterizing the proximity in CHESS. On the other hand, we use the measured values of the delay and loss rate parameters in order to focus on the impact of our bandwidth estimation approach on the proximity characterization.

Therefore, we take the case of file transfer over the TCP protocol. This case can be encountered in the emerging file sharing P2P applications or in the replicated web server context. Applications using TCP are known to form the majority of Internet traffic [22]. For such applications, the optimal peer to select is the one allowing the transfer of the file within the shortest time. We call *latency* the transfer time.

The latency of TCP transfers is known to be a function of diverse network parameters including the available bandwidth, the loss rate, and the round-trip time [23], [24]. Peers are ranked from the standpoint of a certain peer according to an increasing vector of transfer latency. This ranking defines the proximity among peers in the CHESS space. Any other ranking results in a different vector and yields a latency increase. We evaluate in this section the improvement of the TCP latency when the proximity in CHESS is used instead of the delay-based one to perform the ranking of peers from the best to the worst.

To predict the TCP transfer latency, we consider the function PTT (Predicted Transfer Time) that we compute in [24]. This function is the sum of a term that accounts for the slow start phase of TCP and another one that represents the congestion avoidance phase. The function considers the case when a TCP transfer finishes in the slow start with no losses. We omit the window limitation caused by the receiver buffer to allow a better understanding of the impact of path characteristics.

The latency of a TCP transfer depends on the file size. Short transfers are known to be dominated by the slow start phase which is mainly a function of the round-trip time. Long transfers are dominated by the congestion avoidance phase where the available bandwidth and the loss rate figure in addition to the round-trip time. This sensitivity to network parameters makes interesting the problem of peer ranking for applications using large file transfer over TCP.

The enhancement of TCP latency between the proximity in CHESS and the delay-based one is computed as follows. Take a peer p and denote the peer having the rank r in the delay space by $p_d(r)$, i.e., the peer having the r-th smallest RTT on its path to p. Denote by $p_c(r)$ the peer having a rank r in the CHESS space. This peer has, on its path with p, the r-th smallest PTT.

Let $PTT(x, y)$ denote the transfer latency between peer x and peer y. We define the *enhancement* at rank r as the relative error of $PTT(p, p_d(r))$ calculated with respect to $PTT(p, p_c(r))$. More formally, the enhancement is calculated as the following:

$$enhancement(r) = \frac{PTT(p, p_d(r)) - PTT(p, p_c(r))}{PTT(p, p_c(r))} \quad (24)$$

This is realized using the bandwidth values estimated on the paths between a peer p and the 86 other peers using the 40 N-means landmarks. Besides, we use the measured values of the delay RTT and the loss rate P instead of the landmark-based estimated ones in order to focus on the impact of our bandwidth estimation approach on the proximity characterization. We determine the latency enhancement for large files transfer (i.e., $S = \{10MB, 50MB, 100MB\}$) on each path. Then, we average all enhancement values at rank r over the 87 peers. This study allows to evaluate how much the proximity in CHESS outperforms the delay-based one on average at the application level.

We plot in Figure 10 the transfer time enhancement as a function of the rank r for the different file sizes. The enhancement is computed for our three datasets and drawn separately in Figure 10. The figure shows that, for low values of the rank r, the enhancement improves considerably when file size increases. It is between 50% and 200% when the transfer is achieved from the closest peer in the CHESS space. When the rank r increases, the enhancement decreases and it becomes negative at some high ranks. This is due to the fact that the peers having high ranks in the CHESS space (resp. delay space) are those having low ranks in the delay space (resp. CHESS space). The reason is that the paths with farer peers have better performance (e.g., larger available bandwidth) than those with close peers as obtained in Section III. Hence, considering the delay alone for proximity characterization is far from being a good approximation of the proximity characterized in CHESS; this concerns long file transfer applications. The proximity in CHESS provides much better quality even when using the estimated values of the bandwidth parameter.

VII. Conclusions and Perspectives

We introduce in this paper a new notion of proximity in the CHESS space. It consists of characterizing the proximity among peers by considering the path characteristics and application requirements. We motivate the need for this notion by showing that the proximity in the delay space does not automatically lead to a proximity in the bandwidth and loss spaces. The proximity needs to be defined as a function of the metrics impacting the application performance. While network embedding approaches may be convenient for estimating the delay and loss parameters, it is not the case for the bandwidth parameter.

Therefore, we propose a model that infers easily and scalably the bandwidth among peers using the bandwidth of the indirect paths that join them via a set of relay nodes that we call landmarks. Our model depends on the location of the landmarks with respect to the direct path and to the path endpoints. The results show that the required number of landmarks depends on peers' number and distribution. Our experiments show how a number of 40 to 50 landmarks is necessary for estimating the bandwidth among a worldwide distributed set of Planetlab nodes. Finally, the proximity in CHESS, which is determined using our bandwidth estimation model, provides much better quality than that obtained when using the delay proximity for large file transfer applications.

Our perspective concerning the deployment of the landmarks in the real network is that it can be managed by the content service provider. A content service provider can deploy the landmarks according to the distribution of its clients. It may also use an existing set of infrastructural nodes (e.g., DNS servers) as landmarks.

Regarding our future work, we plan to evaluate the profit of characterizing the proximity in CHESS for different applications. On another venue, we will investigate further the problem of scalable bandwidth estimation. In particular, we will study the variability of bandwidth measurements and the persistency of bottlenecks. This is important for determining the measurement frequency of the bandwidth vectors necessary for recomputing and updating the end-to-end bandwidth estimations.

References

[1] F. Dabek, R. Cox, F. Kaashoek, and R. Morris: Vivaldi: A Decentralized Network Coordinate System, ACM Sigcomm, 2004.

[2] B. Gueye, A. Ziviani, M. Crovella, and S. Fdida: Constraint-Based Geolocation of Internet Hosts, ACM IMC, 2004.

[3] E. Ng and H. Zhang: Predicting Internet network distance with coordinates-based approaches, IEEE Infocom, 2002.

[4] L. Tang, and M. Crovella: Virtual Landmarks for the Internet, ACM IMC, 2003.

[5] An open, distributed platform for developing, deploying and accessing planetary-scale network services, see *http://www.planet-lab.org/*.

[6] B. Wong, A. Silvkins, and E. G. Sirer: Meridian: A Lightweight Network Location Service without Virtual Coordinates, ACM Sigcomm, 2005.

[7] P. Francis, S. Jamin, C. Jin, Y. Jin, D. Raz, Y. Shavitt, and L. Zhang: IDMaps: A Global Internet Host Distance Estimation Service, IEEE/ACM Transactions on Networking, 2001.

[8] Y. Shavitt, and T. Tankel: Big-Bang Simulation for Embedding Network Distances in Euclidean Space, IEEE Infocom, 2004.

[9] S. Ratnasamy and M. Handly and R. Karp and S. Shenker: Topologically-Aware Overlay Construction and Server Selection, IEEE Infocom, 2002.

[10] H. Lim, J. Hou, and C. Choi: Constructing Internet Coordinate System based on Delay Measurement, ACM IMC, 2003.

[11] M. Malli, C. Barakat, and W. Dabbous: An Enhanced Scalable Proximity Model, IEEE International workshop on Quality of Service, 2006.

[12] M. Malli, C. Barakat, and W. Dabbous: Landmark-based End-to-End Bandwidth Inference, Infocom student workshop, 2006.

[13] N. Hu, P. Steenkiste: Exploiting Internet Sharing for Large Scale Available Bandwidth Estimation, ACM IMC, 2005.

[14] G. Battista, M. Patrignani, and M. Pizzonia: Computing the types of the relationships betweeen autonomous systems, IEEE Infocom, 2003.

[15] L. Gao: On Inferring Autonomous System Relationships in the Internet, IEEE/ACM Trans. Networking, 2001.

[16] Z. Mao, L. Qiu, J. Wang, and Y. Zhang: On AS-level Path Inference, Sigmetrics, 2005.

[17] L. Peterson, V. Pai, N. Spring, and A. Bavier: Using PlanetLab for Network Research: Myths, Realities, and Best Practices, technical report, 2005.

[18] J. Navratil, L. Cottrell: available at *http://www-iepm.slac.stanford.edu/tools/abing/*, 2004.

[19] J. Navratil, L. Cottrell: ABwE: A Pratical Approach to Available Bandwidth Estimation, PAM, 2003.

[20] J. Navratil: available at *http://www.slac.stanford.edu/ jiri/PLANET*, 2004.

[21] V. Padmanabhan and L. Subramanian: An Investigation of Geographic Mapping Techniques for Internet Hosts, ACM SIGCOMM, 2001.

[22] Cooperative Association for Internet Data Analysis, *http://www.caida.org/*.

[23] N. Cardwell, S. Savage, T. Anderson: Modeling TCP Latency, IEEE Infocom, 2000.

[24] M. Malli, C. Barakat, and W. Dabbous: An Efficient Approach for Content Delivery in Overlay Networks, IEEE CCNC, 2005.

[25] J. Nelder, and R. Meed: A Simplex Method for Function Minimization, Computer Journal, 7:308-313, 1965.

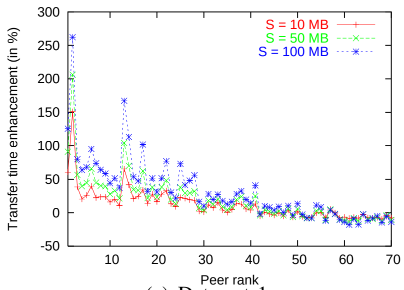
[26] M.Costa, M. Castro, A. Rowstron, and P. Key: PIC: Practical Internet Coordinates for Distance Estimation, ICDCS, 2004.

[27] S. Banerjee, T. Griffin, and M. Pias: The interdomain connectivity of Planetlab nodes, ACM PAM, 2004.

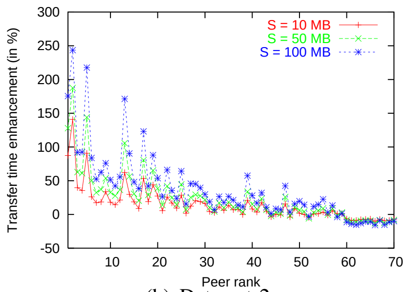
[28] K. Lakshminarayanan, and V. Padmanabhan: Some findings on network performance of broadband hosts, ACM IMC, 2003.

[29] E. Lua, T. Griffin, M. Pias, H. Zheng, and J. Crowcroft, On the Accuracy of Embeddings for Internet Coordinate Systems, ACM IMC, 2005.

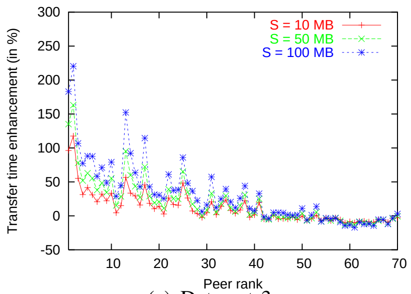
[30] H. Zheng, E. Lua, M. Pias, T. Griffin: Internet Routing Policies and Round-Trip-Times, PAM, 2005.



(a) Data set 1



(b) Data set 2



(c) Data set 3

Fig. 10. Transfer time enhancement when using the proximity in CHESS instead of the delay proximity

[31] M. Malli, C. Barakat, and W. Dabbous: Application-level versus Network-level Proximity, AINTEC, 2005.
[32] Y. Mao, and L. Saul: Modelling distances in large-scale networks by matrix factorization, ACM IMC, 2004.
[33] Y. Mao, and L. Saul, and J. Smith: IDES: An Internet Distance Estimation Service for Large Networks, IEEE JSAC, 2006.