

IDES: An Internet Distance Estimation Service for Large Networks

Yun Mao, *Student Member, IEEE*, Lawrence K. Saul, *Member, IEEE*, and Jonathan M. Smith, *Fellow, IEEE*

Abstract—The responsiveness of networked applications is limited by communications delays, making network distance an important parameter in optimizing the choice of communications peers. Since accurate global snapshots are difficult and expensive to gather and maintain, it is desirable to use sampling techniques in the Internet to predict unknown network distances from a set of partially observed measurements.

This paper makes three contributions. First, we present a model for representing and predicting distances in large-scale networks by *matrix factorization* which can model suboptimal and asymmetric routing policies, an improvement on previous approaches. Second, we describe two algorithms—singular value decomposition and non-negative matrix factorization—for representing a matrix of network distances as the product of two smaller matrices. Third, based on our model and algorithms, we have designed and implemented a scalable system—*Internet Distance Estimation Service (IDES)*—that predicts large numbers of network distances from limited samples of Internet measurements. Extensive simulations on real-world data sets show that IDES leads to more accurate, efficient and robust predictions of latencies in large-scale networks than existing approaches.

Index Terms—Matrix factorization, network distance, network performance.

I. INTRODUCTION

WIDE-AREA distributed applications have evolved considerably beyond the traditional client-server model, in which a client communicates with a single server. In content distribution networks (CDNs), peer-to-peer distributed hash tables (DHTs) [1]–[4], and overlay routing [5], nodes often have a choice of communication peers. Exploiting this choice can greatly improve performance if relevant network distances are known.¹ For example, in a CDN, an optimized client can download Web objects from the particular mirror site to which it has the highest bandwidth. Likewise, in DHT construction, a peer can route lookup requests to the peer (among those that are closer to the target in the virtual overlay network) with the lowest latency in the Internet protocol (IP) underlay network.

Manuscript received October 1, 2005; revised June 21, 2006. The work of L. Saul was supported in part by the National Science Foundation under Grant 0238323. The work of Y. Mao was supported in part by the Defense Advanced Research Projects Agency (DARPA) under Contract F30602-99-1-0512.

The authors are with the Department of Computer and Information Science, University of Pennsylvania, Philadelphia, PA 19104 USA (e-mail: maoy@cis.upenn.edu; lsaul@cis.upenn.edu; jms@cis.upenn.edu).

Digital Object Identifier 10.1109/JSAC.2006.884026

¹Network distance is traditionally known as the round-trip time (RTT) between two hosts. However, in this paper, unless specified otherwise, the definition is generalized to any type of network measurement between nodes, which may or may not be symmetric.

Exact knowledge of network distances, such as that obtained from on-demand network measurements, is expensive and time-consuming to gather and maintain, especially at scale. Thus, a highly promising approach is to construct a model that can *predict* unknown network distances from a set of partially observed measurements [6]–[11].

Many previously proposed models are based on the embedding of host positions in a low-dimensional space, with network distances estimated by Euclidean distances. Such models, however, share certain limitations. In particular, they cannot represent networks with complex routing policies, such as suboptimal routing² or asymmetric routing, since Euclidean distances satisfy the triangle inequality and are inherently symmetric. On the Internet, such routes are quite common [12]–[14], and models that do not take them into account yield inaccurate predictions of network distances.

Our model is based on *matrix factorization* for representing and predicting distances in large-scale networks. The essential idea is to approximate a large matrix whose elements represent pairwise distances by the product of two smaller matrices. Such a model can be viewed as a form of dimensionality reduction. Models based on matrix factorization do not suffer from the limitations of previous work: in particular, they can represent distances that violate the triangle inequality, as well as asymmetric distances. Two algorithms—singular value decomposition (SVD) and non-negative matrix factorization (NMF)—are presented for learning models of this form. We evaluate the advantages and disadvantages of each algorithm for learning compact models of network distances. The basic model was introduced in an earlier paper [15]. The contributions of the present paper are refinements of the model, refined algorithms, and more thorough evaluations of the Internet Distance Estimation Service (IDES). In particular, we address the questions of the impact of both landmark placement and measurement error on IDES performance.

The rest of this paper is organized as follows. Section II reviews previous work based on the low-dimensional embedding of host positions in Euclidean space. Section III presents the model for matrix factorization of network distances. The SVD and NMF algorithms for learning these models from network measurements are presented and evaluated in Section IV. Section V proposes an architecture to estimate distances required by an arbitrary host from low dimensional reconstructions. The architecture is evaluated in Section VI. Finally, Section VII summarizes this paper.

²With suboptimal routing policies, the network distance between two end hosts does not necessarily represent the shortest path in the network. Such routing policies exist widely in the Internet for various technical, political, and economic reasons.

II. NETWORK EMBEDDINGS

One way to predict network distance between arbitrary Internet end hosts is to assign each host a “position” in a finite-dimensional vector space. This can be done at the cost of a limited number of network measurements to a set of well-positioned infrastructure nodes (also known as *landmark* or *beacon* nodes), or other peer nodes. In such a model, a pair of hosts can estimate the network distance between them by applying a distance function to their positions, without direct network measurement. Most previous work on these models has represented the host positions by coordinates in Euclidean space and adopted Euclidean distance as the distance function.

We define the problem formally as follows. Suppose there are N hosts $\mathcal{H} = \{\mathcal{H}_1, \mathcal{H}_2, \dots, \mathcal{H}_N\}$ in the network. The pairwise network distance matrix is a $N \times N$ matrix D , such that $D_{ij} \geq 0$ is the network distance from \mathcal{H}_i to \mathcal{H}_j .

A network *embedding* is a mapping $H : \mathcal{H} \rightarrow \mathbb{R}^d$ such that

$$D_{ij} \approx \hat{D}_{ij} = \|H(\mathcal{H}_i) - H(\mathcal{H}_j)\|, \forall i, j = 1, \dots, N \quad (1)$$

where \hat{D}_{ij} is the estimated network distance from \mathcal{H}_i to \mathcal{H}_j and $H(\mathcal{H}_i)$ is the position coordinate of \mathcal{H}_i as a d -dimensional real vector. We simplify the coordinate notation from $H(\mathcal{H}_i)$ to $\vec{H}_i = (H_{i1}, H_{i2}, \dots, H_{id})$. The network distance between two hosts \mathcal{H}_i and \mathcal{H}_j is estimated by the Euclidean distance of their coordinates

$$\hat{D}_{ij} = \|\vec{H}_i - \vec{H}_j\| = \left(\sum_{k=1}^d (H_{ik} - H_{jk})^2 \right)^{\frac{1}{2}}. \quad (2)$$

The main problem in constructing a network embedding is to compute the position vectors \vec{H}_i for all hosts \mathcal{H}_i from a partially observed distance matrix D . A number of learning algorithms have been proposed to solve this problem, which we describe in the next section.

A. Previous Work

The first work in the network embedding area was done by Ng and Zhang [10], whose Global Network Positioning (GNP) System embedded network hosts in a low-dimensional Euclidean space. Many algorithms were subsequently proposed to calculate the coordinates of network hosts. GNP uses a simplex downhill method to minimize the sum of relative errors

$$\text{total_err} = \sum_i \sum_j \frac{|D_{ij} - \hat{D}_{ij}|}{D_{ij}}. \quad (3)$$

The drawback of GNP is that the simplex downhill method converges slowly, and the final results depend on the initial values of the search. PIC [6] applies the same algorithm to the sum of squared relative errors and studies security-related issues.

Cox *et al.* proposed the Vivaldi algorithm [7], [16], based on an analogy to a network of physical springs. In this approach, the problem of minimizing the sum of errors is related to the problem of minimizing the potential energy of a spring system. Vivaldi has two main advantages: it is a distributed algorithm, and it does not require landmark nodes.

Lim *et al.* [9] and Tang *et al.* [11] independently proposed models based on Lipschitz embeddings and principal compo-

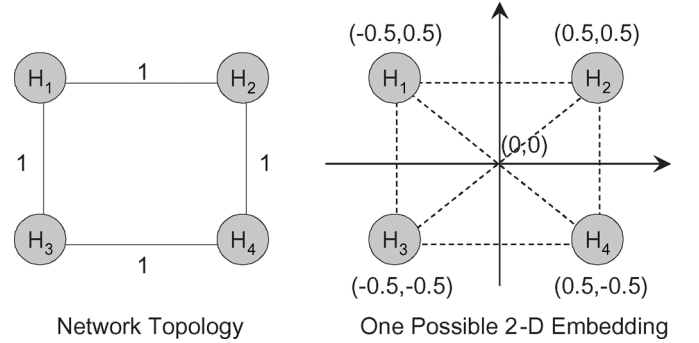


Fig. 1. Four hosts $\mathcal{H}_1 - \mathcal{H}_4$ in a simple network topology.

nent analysis (PCA). These models begin by embedding the hosts in an N -dimensional space, where the coordinates of the host \mathcal{H}_i are given by its distances (D_{i1}, \dots, D_{iN}) to N landmark nodes. This so-called Lipschitz embedding has the property that hosts with similar distances to other hosts are located nearby in the N -dimensional space. To reduce the dimensionality, the host positions in this N -dimensional space are then projected into the d -dimensional subspace of maximum variance by PCA. A linear normalization is used to further calibrate the results, yielding the final host positions $\vec{H}_i \in \mathbb{R}^d$.

B. Limitations

Euclidean distances are inherently symmetric; they also satisfy the triangle inequality. Thus, in any network embedding

$$\begin{aligned} \hat{D}_{ij} &= \hat{D}_{ji} \quad \forall i, j \\ \hat{D}_{ij} + \hat{D}_{jk} &\geq \hat{D}_{ik} \quad \forall i, j, k. \end{aligned}$$

First, the triangle inequality property is inconsistent with observed network distances. On the Internet, studies indicate that as many as 40% of node pairs of real-world data sets have a shorter path through an alternate node [11], [12]. Second, unless the definition of network distances is RTT, the symmetry property also disagrees with the observation. Previous study shows that asymmetric routing is quite common [14]; even for the same link, the upstream and downstream capacities may be very different [13].

In addition to these limitations, low-dimensional embeddings of host positions cannot always model distances in networks where there are pairs of nodes that do not have a direct path between them, even if the distances are symmetric and satisfy the triangle inequality. Fig. 1 illustrates a simple network topology in which four hosts in different autonomous systems are connected with unit distance to their neighbors. An intuitive two-dimensional embedding is also shown. In the given embedding, the estimated distances are $\hat{D}_{14} = \hat{D}_{23} = \sqrt{2}$, but the real distances are $D_{14} = D_{23} = 2$. It is provable that there exists no Euclidean space embedding (of any dimensionality) that can exactly reconstruct the distances in this network. Similar cases arise in networks with tree-like topologies.

III. DISTANCE MATRIX FACTORIZATION

The limitations of previous models lead us to consider a different framework for compactly representing network distances. Suppose that two nearby hosts have similar distances to all the

other hosts in the network. In this case, their corresponding rows in the distance matrix will be nearly identical. More generally, there may be many rows in the distance matrix that are equal or nearly equal to linear combinations of other rows. Recall from linear algebra that an $N \times N$ matrix whose rows are not linearly independent has rank strictly less than N and can be expressed as the product of two smaller matrices. With this in mind, we seek an approximate factorization of the distance matrix, given by

$$D \approx XY^T$$

where X and Y are $N \times d$ matrices with $d \ll N$. From such a model, we can estimate the network distance from \mathcal{H}_i to \mathcal{H}_j by $\hat{D}_{ij} = \vec{X}_i \cdot \vec{Y}_j$, where \vec{X}_i is the i th row vector of the matrix X and \vec{Y}_j is the j th row vector of the matrix Y .

More formally, for a network with distance matrix D_{ij} , we define a *distance matrix factorization* as two mappings

$$\begin{aligned} X : \mathcal{H} &\rightarrow \mathbb{R}^d \\ Y : \mathcal{H} &\rightarrow \mathbb{R}^d \end{aligned}$$

and an approximate distance function computed by

$$\hat{D}_{ij} = X(\mathcal{H}_i) \cdot Y(\mathcal{H}_j).$$

As shorthand, we denote $X(\mathcal{H}_i)$ as \vec{X}_i and $Y(\mathcal{H}_i)$ as \vec{Y}_i , so that we can write the above distance computation as

$$\hat{D}_{ij} = \vec{X}_i \cdot \vec{Y}_j = \sum_{k=1}^d X_{ik} Y_{jk}. \quad (4)$$

Note that in contrast to the model in Section II, which maps each host to one position vector, our model associates *two* vectors with each host. We call \vec{X}_i the *outgoing vector* and \vec{Y}_i the *incoming vector* for \mathcal{H}_i . The estimated distance from \mathcal{H}_i to \mathcal{H}_j is simply the dot product between the outgoing vector of \mathcal{H}_i and the incoming vector of \mathcal{H}_j .

Applying this model of network distances in distributed applications is straightforward. For example, consider the problem of mirror selection. To locate the closest server among several mirror candidates, a client can retrieve the outgoing vectors of the mirrors from a directory server, calculate the dot product of these outgoing vectors with its own incoming vector, and choose the mirror that yields the smallest estimate of network distance (i.e., the smallest dot product).

Our model for representing network distances by matrix factorization overcomes certain limitations of models based on low-dimensional embeddings. In particular, distances computed in this way are not constrained to satisfy the triangle inequality. The main assumption of our model is that many rows in the distance matrix are linearly dependent, or nearly so. This is likely to occur whenever there are clusters of nearby nodes in the network which have similar distances to distant nodes. In this case, the distance matrix D will be well approximated by the product of two smaller matrices.

IV. DISTANCE RECONSTRUCTION

In this section, we investigate how to estimate outgoing and incoming vectors \vec{X}_i and \vec{Y}_i for each host \mathcal{H}_i from the distance

matrix D . We also examine the accuracy of models that approximate the true distance matrix by the product of two smaller matrices in this way.

The distance matrix D can be viewed³ as storing N row vectors in N -dimensional space. Factoring this matrix $D \approx XY^T$ is essentially a problem in linear dimensionality reduction, where Y stores d basis vectors and X stores the linear coefficients that best reconstruct each row vector of D . We present two algorithms for matrix factorization that solve this problem in linear dimensionality reduction.

A. Singular Value Decomposition (SVD)

An $N \times N$ distance matrix D can be factored into three matrices by its SVD, of the form

$$D = USV^T$$

where U and V are $N \times N$ orthogonal matrices and S is an $N \times N$ diagonal matrix with non-negative elements (arranged in decreasing order). Let $A = US^{(1/2)}$ and $B = S^{(1/2)}V$, where $S_{ii}^{(1/2)} = \sqrt{S_{ii}}$. It is easy to see that $AB^T = US^{(1/2)}(VS^{(1/2)})^T = US^{(1/2)}S^{(1/2)}V^T = D$. Thus, SVD yields an exact factorization $D = AB^T$, where the matrices A and B are the same size as D .

We can also use SVD, however, to obtain an approximate factorization of the distance matrix into two smaller matrices. In particular, suppose that only a few of the diagonal elements of the matrix S are appreciable in magnitude. Define the $N \times d$ matrices

$$X_{ij} = U_{ij} \sqrt{S_{jj}}, \quad (5)$$

$$Y_{ij} = V_{ij} \sqrt{S_{jj}} \quad (6)$$

where $i = 1, \dots, N$ and $j = 1, \dots, d$. The product XY^T is a low-rank approximation to the distance matrix D ; if the distance matrix is itself of rank d or less, as indicated by $S_{jj} = 0$ for $j > d$, then the approximation will in fact be exact. The low-rank approximation obtained from SVD can be viewed as minimizing the squared error function

$$\sum_i \sum_j (D_{ij} - \vec{X}_i \cdot \vec{Y}_j)^2 \quad (7)$$

with respect to $X_i \in \mathbb{R}^d$ and $Y_j \in \mathbb{R}^d$. Equations (5) and (6) compute the *global minimum* of this error function.

Matrix factorization by SVD is related to PCA [17] on the row vectors. Principal components of the row vectors are obtained from the orthogonal eigenvectors of their correlation matrix; each row vector can be expressed as a linear combination of these eigenvectors. The diagonal values of S measure the significance of the contribution from each principal component. In previous work on embedding of host positions by PCA, such as ICS [9] and Virtual Landmark [11], the first d rows of the matrix U were used as coordinates for the hosts, while discarding the

³Note that D does not have to be a square matrix of pairwise distances. It can be the distance matrix from one set of N hosts \mathcal{H} to another set of N' hosts \mathcal{H}' , which may or may not overlap with each other. In this case, $X \in \mathbb{R}^{N \times d}$ contains the outgoing vectors for \mathcal{H} and $Y \in \mathbb{R}^{d \times N'}$ contains the incoming vectors for \mathcal{H}' . For simplicity, though, we consider the case $N = N'$ in what follows.

information in the matrices S and V . By contrast, our approach uses U , S , and V to compute outgoing and incoming vectors for each host.

We use the topology in Fig. 1 as an example to show how the algorithm works. The distance matrix is

$$D = \begin{bmatrix} 0 & 1 & 1 & 2 \\ 1 & 0 & 2 & 1 \\ 1 & 2 & 0 & 1 \\ 2 & 1 & 1 & 0 \end{bmatrix}.$$

We obtain the SVD result as

$$U = \begin{bmatrix} -0.5 & 0 & \frac{1}{\sqrt{2}} & 0.5 \\ -0.5 & -\frac{1}{\sqrt{2}} & 0 & -0.5 \\ -0.5 & \frac{1}{\sqrt{2}} & 0 & -0.5 \\ -0.5 & 0 & -\frac{1}{\sqrt{2}} & 0.5 \end{bmatrix}$$

$$S = \begin{bmatrix} 4 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$V = \begin{bmatrix} -0.5 & 0 & -\frac{1}{\sqrt{2}} & -0.5 \\ -0.5 & \frac{1}{\sqrt{2}} & 0 & 0.5 \\ -0.5 & -\frac{1}{\sqrt{2}} & 0 & 0.5 \\ -0.5 & 0 & \frac{1}{\sqrt{2}} & -0.5 \end{bmatrix}.$$

Note that $S_{44} = 0$. Therefore, an exact $d = 3$ factorization exists with

$$X = \begin{bmatrix} -1 & 0 & 1 \\ -1 & -1 & 0 \\ -1 & 1 & 0 \\ -1 & 0 & -1 \end{bmatrix}, Y = \begin{bmatrix} -1 & 0 & -1 \\ -1 & 1 & 0 \\ -1 & -1 & 0 \\ -1 & 0 & 1 \end{bmatrix}.$$

One can verify in this case that the reconstructed distance matrix XY^T is equal to the original distance matrix D .

B. Non-Negative Matrix Factorization (NMF)

Non-negative matrix factorization (NMF) [18] is another form of linear dimensionality reduction that can be applied to the distance matrix D_{ij} . The goal of NMF is to minimize the same error function as in (7), but subject to the constraint that X and Y are non-negative matrices. In contrast to SVD, NMF guarantees that the approximately reconstructed distances are non-negative: $\hat{D}_{ij} \geq 0$. The error function for NMF can be minimized by an iterative algorithm. Compared with gradient descent and the simplex downhill method, however, the algorithm for NMF converges much faster and does not involve any heuristics, such as choosing a step size. The only constraint on the algorithm is that the true network distances must themselves be non-negative, $D_{ij} \geq 0$; this is generally true and holds for all the examples we consider. The algorithm takes as input initial

(random) matrices X and Y and updates them in an alternating fashion. The update rules for each iteration are

$$X_{ia} \leftarrow X_{ia} \frac{(DY)_{ia}}{(XY^T Y)_{ia}}$$

$$Y_{ja} \leftarrow Y_{ja} \frac{(X^T D)_{aj}}{(X^T X Y^T)_{aj}}.$$

It is known that these update rules converge monotonically to stationary points of the error function (7). Our experience shows that two hundred iterations suffice to converge to a local minimum.

One major advantage of NMF over SVD is that it is straightforward to modify NMF to handle missing entries in the distance matrix D . For various reasons, a small number of elements in D may be unavailable. SVD can proceed with missing values if we eliminate the rows and columns in D that contain them, but doing so will leave the corresponding host positions unknown.

NMF can cope with missing values if we slightly change the update rules. Suppose M is a binary matrix, where $M_{ij} = 1$ indicates D_{ij} is known and $M_{ij} = 0$ indicates D_{ij} is missing. The modified update rules are

$$X_{ia} \leftarrow X_{ia} \frac{\sum_k D_{ik} M_{ik} Y_{ka}}{\sum_k (XY^T)_{ik} M_{ik} Y_{ka}} \quad (8)$$

$$Y_{ja} \leftarrow Y_{ja} \frac{\sum_k (X^T)_{ak} D_{kj} M_{kj}}{\sum_k (X^T)_{ak} (XY^T)_{kj} M_{kj}}. \quad (9)$$

These update rules converge to local minima of the error function $\sum_{ij} M_{ij} |D_{ij} - \hat{X}_i \cdot \hat{Y}_j|^2$.

C. Evaluation

We evaluated the accuracy of network distance matrices modeled by SVD and NMF and compared the results to those of PCA from the Lipschitz embeddings used by Virtual Landmark [11] and ICS [9]. We did not evaluate the simplex downhill algorithm used in GNP because while its accuracy is not obviously better than Lipschitz embedding, it is much more expensive, requiring hours of computation on large data sets [11]. Accuracies were evaluated by the modified relative error

$$\text{relative_error} = \frac{|D_{ij} - \hat{D}_{ij}|}{\min(D_{ij}, \hat{D}_{ij})} \quad (10)$$

where the min-operation in the denominator serves to increase the penalty for underestimated network distances.

1) *Data Sets*: We used the following five real-world data sets in simulation. Parts of the data sets were filtered out to eliminate missing elements in the distance matrices (since none of the algorithms except NMF can cope with missing data).

The network distances in the data sets are RTT between pairs of Internet hosts. RTT is symmetric between two end hosts, but it does violate the triangle inequality and also give rise to other effects (described in Section II-B) that are poorly modeled by network embeddings in Euclidean space.⁴

⁴Note that the proposed model can be applied to any type of network measurement between nodes, such as one-way latency, loss rate, and bandwidth, but we have only experimentally validated the model for RTT. Validation on data sets that use different metrics as the distances is still an open problem and will be considered in our future work.

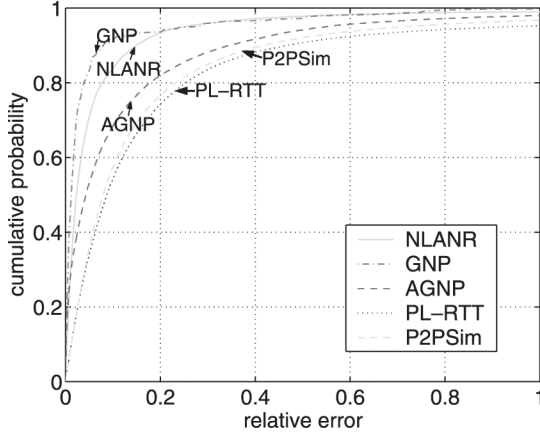


Fig. 2. Cumulative distribution of relative error by SVD over various data sets, $d = 10$.

- **NLANR**: The NLANR Active Measurement Project [19] collects a variety of measurements between all pairs of participating nodes. The nodes are mainly at NSF supported HPC sites, with about 10% outside the U.S. The data set we used was collected on January 30, 2003, consisting of measurements of a 110×110 clique. Each host was pinged once per minute, and network distance was taken as the minimum of the ping times over the day.
- **GNP** and **AGNP**: The GNP project measured minimum round trip time between 19 active sites in May 2001. About 1/2 of the hosts are in North America; the rest are distributed globally. We used GNP to construct a symmetric 19×19 data set and AGNP to construct an asymmetric 869×19 dataset.
- **P2PSim**: The P2PSim project [20] measured a distance matrix of RTTs among about 2000 Internet DNS servers based on the King method [21]. The DNS servers were obtained from an Internet-scale Gnutella network trace.
- **PL-RTT**: Obtained from PlanetLab pairwise ping project [22]. We chose the minimum RTT measured at 3/23/2004 0:00 EST. A 169×169 full distance matrix was obtained by filtering out missing values.

2) *Simulated Results*: Fig. 2 illustrates the cumulative density function (CDF) of relative errors of RTT reconstructed by SVD when $d = 10$, on five RTT data sets. The best result is over GNP data set: more than 90% distances are reconstructed within 9% relative error. This is not too surprising because the GNP data set only contains 19 nodes. However, SVD also works well over NLANR, which has more than 100 nodes: about 90% fraction of distances are reconstructed within 15% relative error. Over P2PSim and PL-RTT data sets, SVD achieves similar accuracy results: 90 percentile relative error is 50%. We ran the same tests on NMF and observed similar results. Therefore, we chose NLANR and P2PSim as two representative data sets for the remaining simulations.

Fig. 3 compares the reconstruction accuracy of three algorithms: matrix factorization by SVD and NMF, and PCA applied to the Lipschitz embedding. The algorithms were simulated over

NLANR and P2PSim data sets. It is shown that NMF has almost exactly the same median relative errors as SVD on both data sets when the dimension $d < 10$. Both NMF and SVD yield much more accurate results than Lipschitz: the median relative error of SVD and NMF is more than five times smaller than Lipschitz when $d = 10$. SVD is slightly better than NMF when d is large. The reason for this may be that the algorithm for NMF is only guaranteed to converge to local minima. Considering that the hosts in the data sets come from all over the Internet, the results show that matrix factorization is a scalable approach to modeling distances in large-scale networks. In terms of maintaining a low-dimensional representation, $d \approx 10$ appears to be a good tradeoff between complexity and accuracy for both SVD and NMF.

V. DISTANCE PREDICTION

The simulation results from the previous section demonstrate that pairwise distances in large-scale networks are well modeled by matrix factorization. In this section, we present the *Internet Distance Estimation Service (IDES)*—a scalable and robust service based on matrix factorization to estimate network distances between arbitrary Internet hosts.

A. Basic Architecture

We classify Internet hosts into two categories: landmark nodes and ordinary hosts. Landmark nodes are a set of well-positioned distributed hosts. The network distances between each of them is available to the *information server* of IDES. We assume that landmarks can measure network distances to others and report the results to the information server. The information server can also measure the pairwise distances via indirect methods without landmark support, e.g., by the King method [21] if the metric is RTT. An ordinary host is an arbitrary end node in the Internet, which is identified by a valid IP address.

Suppose there are m landmark nodes. The first step of IDES is to gather the $m \times m$ pairwise distance matrix D on the information server. Then, we can apply either SVD or NMF algorithm over D to obtain landmark outgoing and incoming vectors \vec{X}_i and \vec{Y}_i in d dimensions, $d < m$, for each host \mathcal{H}_i . As before, we use X and Y to denote the $d \times m$ matrices with \vec{X}_i and \vec{Y}_i as row vectors. Note that NMF can be used even when D contains missing elements.

Now, suppose an ordinary host \mathcal{H}_{new} wants to gather distance information over the network. The first step is to calculate its outgoing vector \vec{X}_{new} and incoming vector \vec{Y}_{new} . To this end, it measures the network distances to and from the landmark nodes. We denote D_i^{out} as the distance to landmark i , and D_i^{in} as the distance from landmark i to the host. Ideally, we would like the outgoing and incoming vectors to satisfy $D_i^{\text{out}} = \vec{X}_{\text{new}} \cdot \vec{Y}_i$ and $D_i^{\text{in}} = \vec{X}_i \cdot \vec{Y}_{\text{new}}$. The solution with the least squares error is given by

$$\vec{X}_{\text{new}} = \arg \min_{\vec{U} \in \mathbb{R}^d} \sum_{i=1}^m \left(D_i^{\text{out}} - \vec{U} \cdot \vec{Y}_i \right)^2 \quad (11)$$

$$\vec{Y}_{\text{new}} = \arg \min_{\vec{U} \in \mathbb{R}^d} \sum_{i=1}^m \left(D_i^{\text{in}} - \vec{X}_i \cdot \vec{U} \right)^2. \quad (12)$$

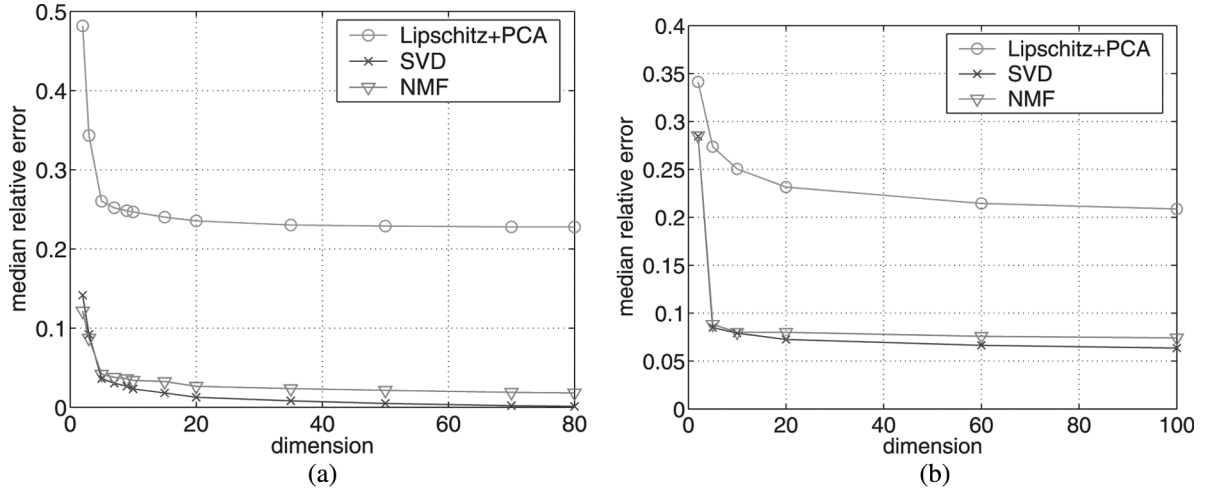


Fig. 3. Reconstruction error comparison of SVD, NMF, and Lipschitz over NLNR and P2PSim data set. (a) Comparison over NLNR data set. (b) Comparison over P2PSim data set.

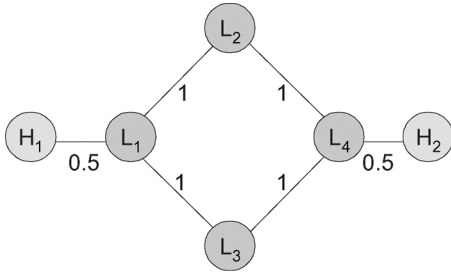


Fig. 4. Four landmark nodes $L_1 - L_4$ and two ordinary hosts \mathcal{H}_1 and \mathcal{H}_2 interconnected by a simple network topology.

The global minima of these error functions, computed by simple matrix operations, have the closed form

$$\vec{X}_{\text{new}} = (D^{\text{out}} Y)(Y^T Y)^{-1} \quad (13)$$

$$\vec{Y}_{\text{new}} = (D^{\text{in}} X)(X^T X)^{-1}. \quad (14)$$

Equations (13) and (14) assume that the optimizations are unconstrained. Alternatively, one can impose non-negativity constraints on \vec{X}_{new} and \vec{Y}_{new} ; this will guarantee that the predicted distances are themselves nonnegative (assuming that the landmark distance matrix was also modeled by NMF). The least squared error problems in (11) and (12) can be solved with non-negativity constraints, but the solution is somewhat more complicated. Our simulation results did not reveal any significant difference between the prediction accuracies of least squares solutions with and without non-negativity constraints; thus, in what follows, we focus on the simpler unconstrained solutions in (13) and (14).

We give a simple example of this procedure in Fig. 4. The network is an enlarged version of the network in Fig. 1, with the four original nodes serving as landmarks and two new nodes introduced as ordinary hosts. The first step is to measure interlandmark distances and calculate landmark incoming and outgoing

vectors. We used SVD to factor the landmark distance matrix in this example. The result is the same as the example in Section IV

$$X = \begin{bmatrix} -1 & 0 & 1 \\ -1 & -1 & 0 \\ -1 & 1 & 0 \\ -1 & 0 & -1 \end{bmatrix}, Y = \begin{bmatrix} -1 & 0 & -1 \\ -1 & 1 & 0 \\ -1 & -1 & 0 \\ -1 & 0 & 1 \end{bmatrix}.$$

Note that SVD can be substituted by NMF and the following steps are identical.

Second, we measure the distance vectors for the ordinary hosts: $D^{\text{out}} = D^{\text{in}} = [0.5 \ 1.5 \ 1.5 \ 2.5]$ for ordinary host \mathcal{H}_1 . According to (13) and (14), $\vec{X}_{\mathcal{H}_1} = [-1.5 \ 0 \ 1]$ and $\vec{Y}_{\mathcal{H}_1} = [-1.5 \ 0 \ -1]$. Similarly, we obtain the distance vector of \mathcal{H}_2 as $[2.5 \ 1.5 \ 1.5 \ 0.5]$, and calculate its outgoing and incoming vectors: $\vec{X}_{\mathcal{H}_2} = [-1.5 \ 0 \ -1]$ and $\vec{Y}_{\mathcal{H}_2} = [-1.5 \ 0 \ 1]$. One can verify that distances between ordinary hosts and landmarks are exactly preserved. The distance between two ordinary hosts is not measured, but can be estimated as $\vec{X}_{\mathcal{H}_1} \cdot \vec{Y}_{\mathcal{H}_2} = \vec{X}_{\mathcal{H}_2} \cdot \vec{Y}_{\mathcal{H}_1} = 3.25$, while the real network distance is 3.

B. Optimization

The basic architecture requires an ordinary host to measure network distances to all landmarks, which limits the scalability of IDES. Furthermore, if some of the landmark nodes experience transient failures or a network partition, an ordinary host may not be able to retrieve the measurements it needs to solve (13) and (14).

To improve the scalability and robustness of IDES, we propose a relaxation to the basic architecture: an ordinary host \mathcal{H}_{new} only has to measure distances to a set of k nodes with precomputed outgoing and incoming vectors. The k nodes can be landmark nodes, or other ordinary hosts that have already computed their vectors. Suppose the outgoing vectors of those k nodes are $\vec{X}_1, \vec{X}_2, \dots, \vec{X}_k$ and the incoming vectors are $\vec{Y}_1, \vec{Y}_2, \dots, \vec{Y}_k$. We measure D_i^{out} and D_i^{in} as the distance from and to the i th node, for all $i = 1, \dots, k$. Calculating the new

vectors \vec{X}_{new} and \vec{Y}_{new} for \mathcal{H}_{new} is done by solving the least squares problems

$$\vec{X}_{\text{new}} = \arg \min_{\vec{U} \in \mathbb{R}^d} \sum_{i=1}^k \left(D_i^{\text{out}} - \vec{U} \cdot \vec{Y}_i \right)^2 \quad (15)$$

$$\vec{Y}_{\text{new}} = \arg \min_{\vec{U} \in \mathbb{R}^d} \sum_{i=1}^k \left(D_i^{\text{in}} - \vec{X}_i \cdot \vec{U} \right)^2. \quad (16)$$

The solution is exactly the same form, as described in (13) and (14). The constraint $k \geq d$ is necessary (and usually sufficient) to ensure that the problem is not singular. In general, larger values of k lead to better prediction results, as they incorporate more measurements of network distances involving \mathcal{H}_{new} into the calculation of the vectors \vec{X}_{new} and \vec{Y}_{new} .

We use the topology in Fig. 4 again to demonstrate how the system works. As in the basic architecture, the first step is to measure interlandmark distances and calculate landmark outgoing and incoming vectors. Second, the ordinary host \mathcal{H}_1 measures the distances to L_1 , L_2 , and L_3 as [0.5 1.5 1.5]. By (13) and (14), the vectors are $\vec{X}_{\mathcal{H}_1} = [-1.5 \ 0 \ 1]$ and $\vec{Y}_{\mathcal{H}_1} = [-1.5 \ 0 \ -1]$. Note that we did not measure the distance between \mathcal{H}_1 and L_4 , but it can be estimated as $\vec{X}_{\mathcal{H}_1} \cdot \vec{Y}_{L_4} = [-1.5 \ 0 \ 1] \cdot [-1 \ 0 \ 1] = 2.5$, which is in fact the true distance. Finally, the ordinary host \mathcal{H}_2 measures the distances to L_2 , L_4 , and \mathcal{H}_1 as [1.5 0.5 3]. Because all of them already have precomputed vectors, \mathcal{H}_2 can compute its own vectors by (13) and (14). The results are $\vec{X}_{\mathcal{H}_2} = [-1.4 \ 0.1 \ -0.9]$ and $\vec{Y}_{\mathcal{H}_2} = [-1.4 \ -0.1 \ 0.9]$. The distances between ordinary host \mathcal{H}_2 and L_1/L_3 are not measured directly, but can be estimated as $\vec{X}_{\mathcal{H}_2} \cdot \vec{Y}_{L_1} = [-1.4 \ 0.1 \ -0.9] \cdot [-1 \ 0 \ -1] = 2.3$ and $\vec{X}_{\mathcal{H}_2} \cdot \vec{Y}_{L_3} = [-1.4 \ 0.1 \ -0.9] \cdot [-1 \ -1 \ 0] = 1.3$.

This example illustrates that even without measurement to all landmarks, the estimated distances can still be accurate. In this example, most of the pairwise distances are exactly preserved; the maximum relative error is 15% when predicting the distance between \mathcal{H}_2 and L_2 . In the example, the load is well distributed among landmarks. As shown in Fig. 5, distances to L_2 are only measured twice during this estimation procedure. Such a scheme allows IDES to scale to a large number of ordinary hosts and landmarks. It is also robust against partial landmark failures.

C. Landmark Selection

The system performance depends on the positioning of the landmarks. Ill-positioned landmarks can significantly reduce the accuracy of estimated network distances. For example, as a worst case, imagine that all the landmarks are very close to each other. Then, the pairwise distance matrix D will be close to the zero matrix, and modeling D by matrix factorization will not capture any information about long-range distances or the global network topology. Given a set of nodes and their pairwise network distances, we propose the following four landmark selection algorithms and evaluate them in the next section.

- **Random:** We randomly choose landmarks by uniformly sampling (without replacement) from the set of all candidate nodes. In this scheme, each node has the same proba-

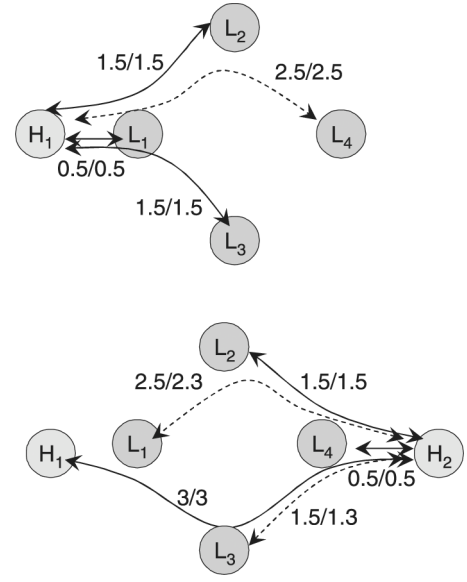


Fig. 5. Learning outgoing and incoming vectors for two ordinary hosts. Solid lines indicate that real network measurement is conducted. Each edge is annotated with (real network distance/estimated distance).

bility to be chosen as a landmark, and no information about pairwise distances is required.

- **k -means:** The k -means algorithm [23] is a well-known iterative procedure for detecting k clusters in multivariate data. We adapt the algorithm to find clusters in the space of host nodes; the host nodes closest to the cluster centroids are then chosen as landmark nodes. Since only distances between host nodes are initially specified, it is necessary to represent the host nodes as points in a vector space before applying the k -means algorithm. We choose the Lipschitz embedding method: each node \mathcal{H}_i is represented by the N -dimensional vector whose elements are its distances (D_{i1}, \dots, D_{iN}) to the other nodes in the network. Note that Euclidean distances in this representation do not preserve the distances originally specified by the matrix D , as discussed in Section II.
- **Spectral clustering:** Spectral clustering [24] is another well-known clustering procedure. Unlike k -means, it can cluster objects directly based on their pairwise distances, without requiring an intermediate vector space representation. Moreover, the main step of spectral clustering is an eigenvector computation, not an iterative optimization with potentially spurious local minima, as in k -means. In general, spectral clustering is more robust than k -means, but also more computationally expensive. We use the algorithm in [24] to assign nodes to k different clusters and choose the nodes closest to each cluster centroid as landmark nodes.
- **Maximum distance:** Finally, we use a simple greedy heuristic to identify landmarks whose summed pairwise distances are approximately maximized. (An optimal algorithm would be NP-hard.) Our algorithm works as follows: 1) initially, we choose one node at random as the first landmark; 2) from the remaining nodes, we choose the node with maximum average distance to the existing

landmarks as the next landmark; and 3) we repeat step 2) until enough landmarks have been chosen.

VI. EVALUATION

In this section, we evaluate IDES, using SVD and NMF algorithms to learn models of network distances, and compare them to the GNP [10] and ICS [9] systems.

The experiments were performed on a Dell Dimension 4600 with Pentium 4 3.2 GHz CPU, 2 GB RAM. The GNP implementation was obtained from the official GNP software release written in C. We implemented IDES and ICS exactly as described in [9] in MatLab 6.0. Please refer to Section II-A for details of the GNP and ICS systems.

We identify four evaluation criteria.

- **Efficiency:** We measure efficiency by the total running time required by a system to build its model of network distances between all landmark nodes and ordinary hosts.
- **Accuracy:** The prediction error between D_{ij} and \hat{D}_{ij} should be small. We use the modified relative error function in (10) to evaluate accuracy, which is also used in GNP and Vivaldi. Note that predicted distances are computed between ordinary hosts that have not conducted any network measurements of their distance. Predicted distance errors are different than reconstructed distance errors (where actual network measurements are conducted). Evaluations based on other proposed error functions [25] will be considered in our future work.
- **Scalability:** The storage requirements are $O(d)$ for models based on network embeddings (with one position vector for each host) and matrix factorizations (with one incoming and outgoing vector for each host). In large-scale networks, the number of hosts N is very large. The condition $d \ll N$ allows the model to scale, assuming that reasonable accuracy of predicted distances is maintained. Also, to support multiple hosts concurrently, it is desirable to distribute the load—for instance, by only requiring distance measurements to partial sets of landmarks.
- **Robustness:** A robust system should be resilient against host failures, temporary network partitioning, and measurement errors. In particular, partial failure of landmark nodes should not prevent the system from building models of network distances.

A. Efficiency and Accuracy

We use three data sets for evaluating accuracy and efficiency.

- **GNP:** 15 out of 19 nodes in the symmetric data set were selected as landmarks. The rest of the 4 nodes and the 869 nodes in the AGNP data set were selected as ordinary hosts. Prediction accuracy was evaluated on 869×4 pairs of hosts.
- **NLANR:** 20 out of 110 nodes were selected randomly as landmarks. The remaining 90 nodes were treated as ordinary hosts. The prediction accuracy was evaluated on 90×90 pairs of hosts.
- **P2PSim:** 20 out of 1143 nodes were selected randomly as landmarks. The remaining 1123 nodes were treated as ordinary hosts. The prediction accuracy was evaluated on 1123×1123 pairs of hosts.

TABLE I

EFFICIENCY COMPARISON ON IDES, ICS, AND GNP OVER FOUR DATA SETS

data set	IDES/SVD	IDES/NMF	ICS	GNP
GNP	0.10s	0.12s	0.02s	1min 19s
NLANR	0.01s	0.02s	0.01s	4min 44s
P2PSim	0.16s	0.17s	0.03s	2min 30s

Although deliberate placement of landmarks might yield more accurate results, we chose the landmarks randomly since, in general, they may be placed anywhere on the Internet. We present the study of landmark placement effect in Section VI-C. To ensure fair comparisons, we used the same set of landmarks for all four algorithms. We also repeated the simulation several times, and no significant differences in results were observed from one run to the next.

Table I illustrates the running time comparison between IDES, ICS, and GNP. GNP is much less efficient than the IDES and ICS. This is because GNP uses simplex downhill method, which converges slowly to local minima. Both IDES and ICS have running time less than 1 s, even when the data sets contain thousands of nodes. It is possible to reduce the running time of GNP by sacrificing the accuracy, but the parameters are hard to tune, which is another drawback of simplex downhill method.

Fig. 6 plots the CDF of prediction errors for IDES using SVD, IDES using NMF, ICS, and GNP over the three data sets, respectively. In Fig. 6(a), the GNP system is the most accurate system for the GNP data set. IDES using SVD and NMF are as accurate as GNP for 70% of the predicted distances. The GNP data set is somewhat atypical, however, in that the predicted distance matrix has many more columns (869) than rows [4]. Fig. 6(b) and (c) depicts the CDF of prediction errors over NLANR and P2PSim data sets, which are more typical. In both cases, IDES has the best prediction accuracy. On the NLANR data set, IDES yields better results than GNP and ICS: the median relative error of IDES using SVD is only 0.03. Its 90th percentile relative error is about 0.23. The accuracy is worse for all three systems in P2PSim data set than in NLANR data set. However, IDES (with either SVD or NMF) is still the most accurate system among the three. The better prediction results on the NLANR data set may be due to the fact that 90% of the hosts in NLANR are in North America and the network distances, computed from minimum RTT over a day, are not affected much by queueing delays and route congestion. These properties make the data set more uniform, and therefore, more easily modeled by a low-dimensional representation.

B. Scalability and Robustness

In the previous section, we showed that IDES can accurately model the network distances in low dimensions $d \leq 10$, which is fundamental to make the system scale to large-scale networks. In this section, we study the impact of partially observed landmarks and network measurement errors on the accuracy of IDES.

1) *Partially Observed Landmarks:* Measuring the distances to only a subset of landmark nodes reduces the overall load and allows the system to support more ordinary hosts concurrently. It also makes the system robust to partial landmark failures.

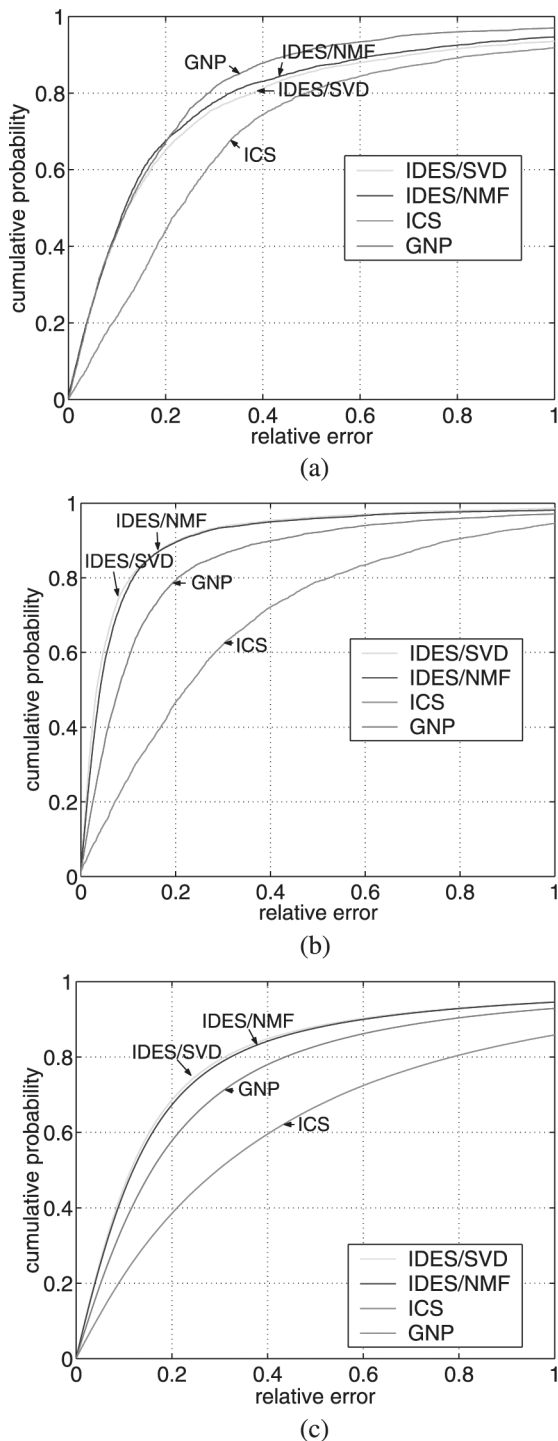


Fig. 6. Accuracy comparison on IDES using SVD and NMF, ICS, and GNP, $d = 8$. (a) CDF of relative error over GNP data set, 15 landmarks. (b) CDF of relative error over NLANR data set, 20 landmarks. (c) CDF of relative error over P2PSim data set, 20 landmarks.

We simulated partially observed landmark scenarios in IDES using SVD to model partial distance matrices from the NLANR and P2PSim data sets. For each data set, we experimented with two settings: 20 random landmarks and 50 random landmarks. The simulation results are shown in Fig. 7. The x axis indicates the fraction of unobserved landmarks. The unobserved landmarks for each ordinary host were independently generated at

random. When the number of landmarks is less than twice the model dimensionality d , the accuracy appears sensitive to the fraction of unobserved landmarks. However, as the number of landmarks increases, the system tolerates more failure: for example, not observing 40% of the landmarks has little impact on the system accuracy when 50 landmarks are used in the test.

2) *Measurement Errors*: In previous simulations, we assume that all the measurement results from the data sets are accurate. However, in the real world, measurement may contain various kinds of errors. For example, a temporary route change may result unstable measurement results; a very high load node may respond the ICMP ECHO or UDP packet with a higher delay⁵; a transparent Web proxy may intercept TCP SYN packets and respond with SYN/ACK packets to make TCP probing results inaccurate. A robust system should be resilient against a small amount of measurement anomalies and still yield reasonably accurate results.

To this end, we studied how well IDES and ICS work by simulating measurement errors. We first assumed that the original data sets do not contain erroneous results. Then, among the $n \times (n - 1)$ links in the distance matrix D , we randomly picked $pn(n - 1)$ links, where $0 < p < 1$, and increased the RTTs of these links to λ times of the original values. This is to say that, in the following simulation, the measured network distances could be larger than the actual values as much as $\lambda - 1$ times. We simulated the IDES system using SVD and ICS, respectively. Twenty landmarks were chosen by the random algorithm, and the dimension used was $d = 10$. Note that when calculating the estimation errors, we compared the estimated results to the actual network distances (i.e., the original data without injected errors). We ran the simulation over the NLANR and P2P data sets, repeated the test 100 times, and report the average median estimation errors in Fig. 8. Not surprisingly, in the IDES system, the amount of relative estimation errors increases when the amount of inaccurately measured links increases. The higher the degree of measurement error λ is, the faster the estimation error increases along with p . However, even when $p = 0.05$ and $\lambda = 3$, i.e., 5% of the network links are subject to two times more measurement errors, IDES is still more accurate than the case of ICS where no link suffers any measurement error. In ICS, the accuracy is not affected much when λ is low. However, we noticed significant estimation error when $p = 0.15$ and $\lambda = 10$ over both data sets, which means that ICS is not as robust as IDES when the measurement error range is large, the reason for which deserves further investigation.

C. Landmark Selection

In this section, we use simulation to evaluate the impacts of different landmark selection algorithms on the accuracy of IDES. Except for the random algorithm, we assume the pairwise network distances among all the nodes are given when choosing the landmarks. Although the assumption is somewhat unrealistic in very large-scale networks, understanding the relationship between landmark distribution and prediction accuracy is important.

⁵The data set from PlanetLab pairwise ping project [22] show that some of the RTTs are longer than 2000 ms, which is unlikely to happen in today's Internet. We suspect this is due to the fact that the PlanetLab nodes are often overloaded.

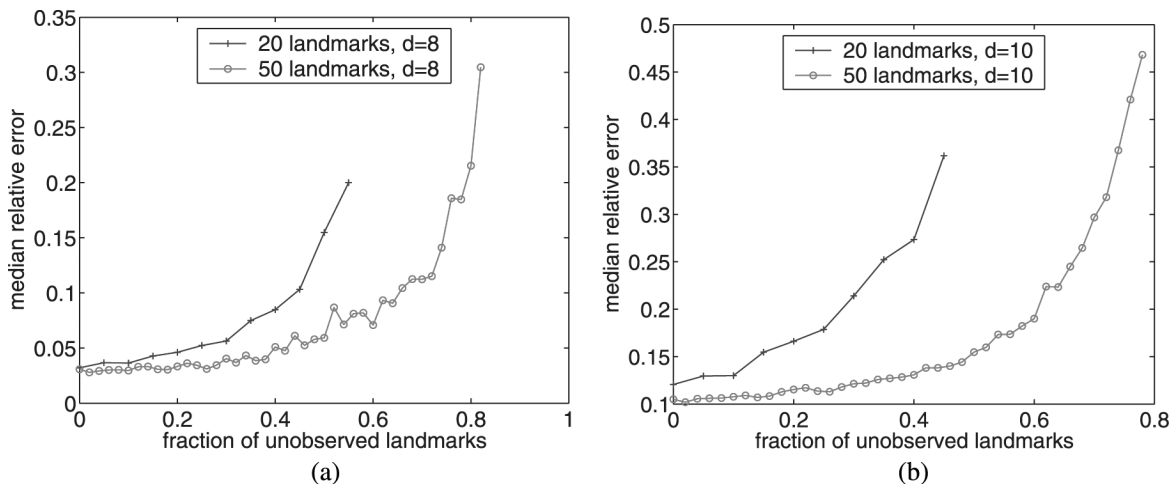


Fig. 7. The correlation between accuracy and landmark failures on IDES using SVD algorithm. a) Over NLANR data set. b) Over P2PSim data set.

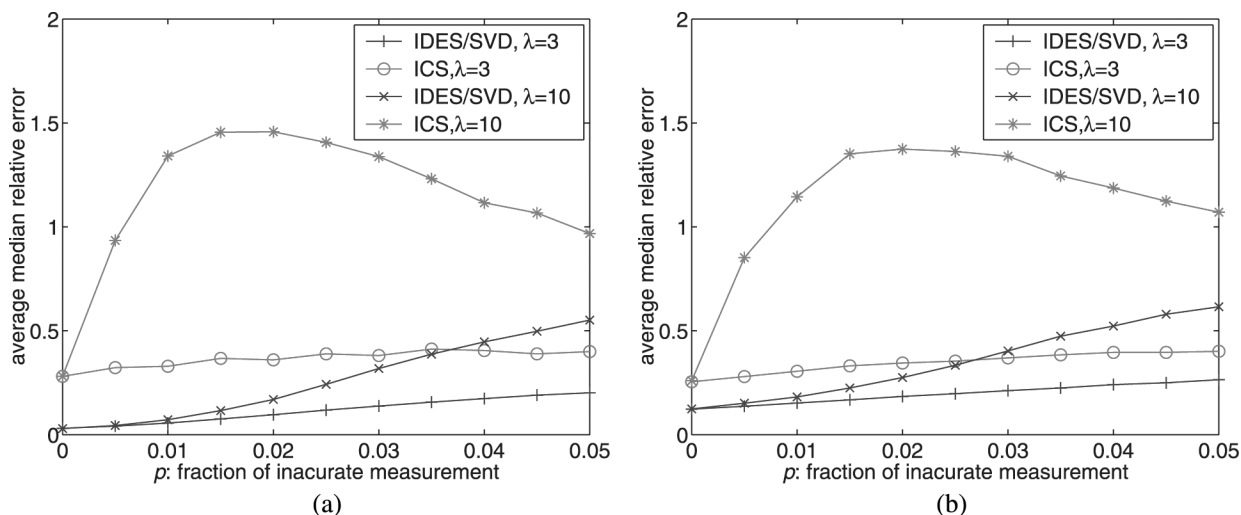


Fig. 8. The impact of measurement errors on the accuracy of IDES and ICS. In each test, there is p fraction of links and are subject to measurement errors. a) Over NLANR data set. b) Over P2PSim data set.

We studied the four algorithms proposed in Section V-C, the random algorithm, the k -means algorithm, the spectral clustering algorithm, and the maximum distance algorithm. Each of them was evaluated over three data sets (NLANR, P2PSim, and PL-RTT). After using a particular algorithm to select a predefined number of landmarks (20 in the simulation), we ran IDES using SVD with $d = 10$ to predict the network distances between ordinary nodes, and recorded the 90th percentile relative error of the predicted distances. For each landmark selection algorithm and data set, we repeated the test 100 times, and reported the average and standard deviation of the 90th percentile relative error in Fig. 9(a) and (b), respectively. From the figures, we can see that the average errors of the random, k -means and spectral clustering algorithms are quite close, and it is consistent over the three data sets. However, maximum distance algorithm is much worse: for example, the average 90th percentile relative error of the maximum algorithm is about five times larger than the errors of the other three algorithms over the P2PSim data set. This indicates that the maximum distance algorithm is not appropriate for landmark selection in IDES system. From Fig. 9(b), we can see that the standard deviation

of the 90th percentile relative error of the random algorithm is larger than the k -means and spectral clustering algorithms. This is due to the fact that random selections do not necessarily prevent “bad” landmark selections. Therefore, the stability of yielded accuracy of the random algorithm is lower than the two clustering based algorithms. The k -means and spectral clustering algorithms slightly outperform each other in different data sets. We conclude that the maximum distance algorithm is not a good choice of landmark selection; the random algorithm, although not as stable as the clustering-based algorithms, performs reasonably well and has the least time complexity without assuming a given distance matrix. A previous study on virtual network coordinates also showed that random landmark selection is fairly effective if more than 20 landmarks are employed [26].

VII. SUMMARY

In this paper, we have presented a model based on matrix factorization for predicting network distances between arbitrary Internet hosts. Our model imposes fewer constraints on network distances than models based on low dimensional embed-

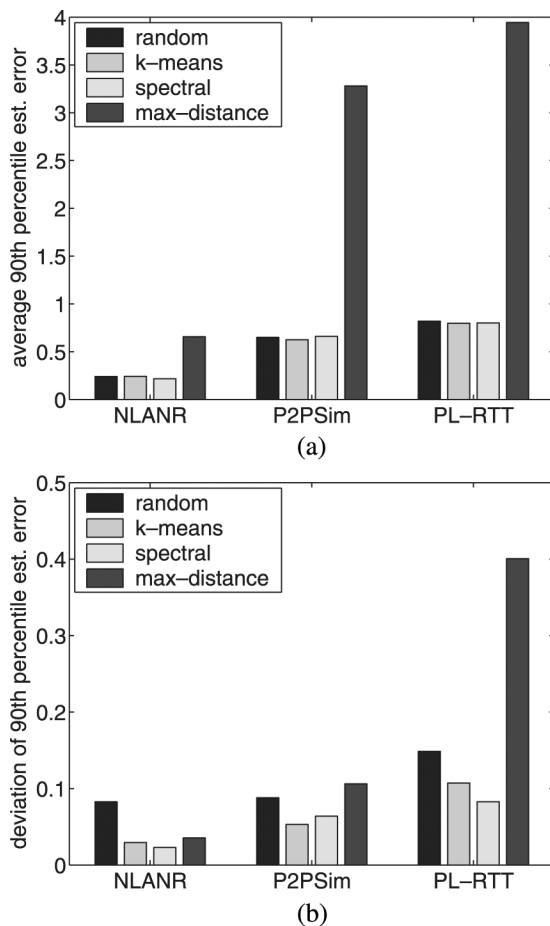


Fig. 9. Performance comparison on the landmark selection algorithms (random, k-means, spectral clustering and maximum distance) over the datasets of NLANR, P2PSim, and PL-RTT. 20 landmarks are chosen. $d = 10$. (a) Comparison of the average 90th percentile estimation. (b) Comparison of the standard deviation of 90th percentile estimation errors.

dings; in particular, it can represent distances that violate the triangle inequality. Such a model is more suitable for modeling the topology and complex routing policies on the Internet. Based on this model, we proposed the IDES system and two learning algorithms, SVD and NMF, for factoring matrices of network distances between arbitrary Internet hosts. Simulations on real-world data sets have shown that IDES is computationally efficient, scalable to large-scale networks, more accurate than previous models, and resilient to temporary landmark failures.

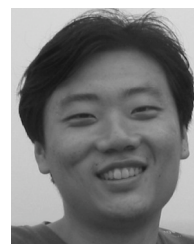
ACKNOWLEDGMENT

The authors are grateful to F. Dabek from MIT for sharing the P2PSim data set.

REFERENCES

- [1] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker, "A scalable content-addressable network," in *Proc. ACM SIGCOMM Conf.*, San Diego, CA, Aug. 2001.
- [2] A. Rowstron and P. Druschel, "Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems," in *Proc. IFIP/ACM Int. Conf. Distrib. Syst. Platforms (Middleware)*, Nov. 2001, pp. 329–350.

- [3] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, "Chord: A scalable peer-to-peer lookup service for Internet applications," in *Proc. ACM SIGCOMM Conf.*, San Diego, CA, Aug. 2001.
- [4] B. Y. Zhao, J. D. Kubiatowicz, and A. D. Joseph, "Tapestry: An infrastructure for fault-tolerant wide-area location and routing," Univ. California at Berkeley, Berkeley, CA, Tech. Rep. CSD-01-1141, Apr. 2001.
- [5] D. G. Andersen, H. Balakrishnan, M. F. Kaashoek, and R. Morris, "Resilient overlay networks," in *Proc. 18th ACM SOSP*, 2001.
- [6] M. Costa, M. Castro, A. Rowstron, and P. Key, "PIC: practical Internet coordinates for distance estimation," in *Proc. 24th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Tokyo, Japan, Mar. 2004, pp. 178–187.
- [7] F. Dabek, R. Cox, F. Kaashoek, and R. Morris, "Vivaldi: A decentralized network coordinate system," in *Proc. ACM SIGCOMM Conf.*, Aug. 2004.
- [8] 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 Trans. Netw.*, pp. 525–540, Oct. 2001.
- [9] H. Lim, J. Hou, and C.-H. Choi, "Constructing Internet coordinate system based on delay measurement," in *Proc. Internet Measure. Conf.*, Oct. 2003, pp. 513–525.
- [10] T. S. E. Ng and H. Zhang, "Predicting Internet network distance with coordinates-based approaches," in *Proc. INFOCOM*, New York, Jun. 2002, pp. 170–179.
- [11] L. Tang and M. Crovella, "Virtual landmarks for the Internet," in *Proc. Internet Measure. Conf.*, Oct. 2003.
- [12] S. Banerjee, T. G. Griffin, and M. Pias, "The interdomain connectivity of planetlab nodes," in *Proc. 5th Annu. Passive Active Measure. Workshop*, Antibes Juan-les-Pins, France, Apr. 2004.
- [13] K. Lakshminarayanan and V. Padmanabhan, "Some findings on the network performance of broadband hosts," in *Proc. Internet Measure. Conf.*, Oct. 2003.
- [14] V. Paxson, "End-to-end routing behavior in the Internet," *IEEE/ACM Trans. Netw.*, vol. 5, no. 5, pp. 601–615, Oct. 1997.
- [15] Y. Mao and L. K. Saul, "Modeling distances in large-scale networks by matrix factorization," in *Proc. 2004 ACM SIGCOMM Internet Measure. Conf.*, Taormina, Italy, Oct. 2004, pp. 278–287.
- [16] R. Cox, F. Dabek, F. Kaashoek, J. Li, and R. Morris, "Practical, distributed network coordinates," in *Proc. HotNets-II*, Cambridge, MA, Nov. 2003.
- [17] I. T. Jolliffe, *Principal Component Analysis*. New York: Springer-Verlag, 1986.
- [18] D. D. Lee and H. S. Seung, "Algorithms for non-negative matrix factorization," in *Proc. Neural Inf. Process. Syst.*, 2000, pp. 556–562.
- [19] The NLANR Active Measurement Project. [Online]. Available: <http://amp.nlanr.net/active/>
- [20] The P2PSim Project. [Online]. Available: <http://www.pdos.lcs.mit.edu/p2psim>
- [21] K. P. Gummadi, S. Saroiu, and S. D. Gribble, "King: Estimating latency between arbitrary internet end hosts," in *Proc. SIGCOMM Internet Measure. Workshop*, Marseille, France, Nov. 2002.
- [22] J. Stribling, "All pairs of ping data for PlanetLab." [Online]. Available: http://www.pdos.lcs.mit.edu/~strib/pl_app
- [23] C. M. Bishop, *Neural Networks for Pattern Recognition*. Oxford, U.K.: Oxford Univ. Press, 1995.
- [24] A. Y. Ng, M. I. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," in *NIPS*, T. G. Dietterich, S. Becker, and Z. Ghahramani, Eds. Cambridge, MA: MIT Press, 2001, pp. 849–856.
- [25] E. K. Lua, T. Griffin, M. Pias, H. Zheng, and J. Crowcroft, "On the accuracy of embeddings for Internet coordinate systems," in *Proc. ACM SIGCOMM Internet Measure. Conf.*, Berkeley, CA, Oct. 2005.
- [26] L. Tang and M. Crovella, "Geometric exploration of the landmark selection problem," in *Proc. 5th Annu. Passive Active Measure. Workshop*, Antibes Juan-les-Pins, France, Apr. 2004.



Yun Mao (S'05) received the B.E. and M.E. degrees in computer science from Tsinghua University, Beijing, China, in 2000 and 2002, respectively. He is currently working towards the Ph.D. degree at the Department of Computer and Information Science, University of Pennsylvania, Philadelphia.

From Fall 2005 to Spring 2006, he was a Visiting Research Fellow at the Department of Computer Science, University of Texas at Austin. His current main research interests are real-world networked systems, including the aspects of performance, mobility, reliability, scalability, and manageability.

Mr. Mao received the Chair's Award from the University of Pennsylvania as one of a small number of particularly outstanding doctoral students in 2002.



Lawrence K. Saul (M'04) received the A.B. degree in physics from Harvard University, Cambridge, MA, in 1990 and the Ph.D. degree in physics from the Massachusetts Institute of Technology (MIT), Cambridge, in 1994.

He stayed at MIT for two more years as a Postdoctoral Fellow at the Center for Biological and Computational Learning, then joined the research wing at AT&T Laboratories, Murray Hill, NJ, and later at Florham Park, NJ. In 2002, he joined the faculty of the Department of Computer and Information Science, University of Pennsylvania. In 2006, he joined the faculty of the Department of Computer Science and Engineering, University of California at San Diego. His main research interests lie in machine learning, pattern recognition, voice processing, and auditory computation.

Dr. Saul was named as one of 100 Top Young Innovators by Technology Review in 1999. In 2001, he was a founding member of the Editorial Board for the *Journal of Machine Learning Research*.

Dr. Saul was named as one of 100 Top Young Innovators by Technology Review in 1999. In 2001, he was a founding member of the Editorial Board for the *Journal of Machine Learning Research*.



Jonathan M. Smith (S'86–M'89–SM'93–F'01) received the Ph.D. degree from Columbia University, New York, in 1989.

He is the Olga and Alberico Pompa Professor of Engineering and Applied Science at the University of Pennsylvania, and a Professor in the Department of Computer and Information Science, University of Pennsylvania, where he has been since 1989.

At the University of Pennsylvania, his research has been focused on advanced communication and computer networking systems such as gigabit networks,

where his impact influenced research and commercial software and hardware, and resulted in several widely cited U.S. patents. He was previously at Bell Telephone Laboratories and Bellcore, where he focused on UNIX internals, tools and distributed computing technology, and was a member of a technology transfer team for computer security. He has consulted extensively for industry and government. He spent much of 1997 on sabbatical at the University of Cambridge, and is currently on leave from the University of Pennsylvania and is at the Defense Advanced Research Projects Agency (DARPA). His current research interests are programmable network infrastructures and computer security.

Dr. Smith is a member of the Association for Computing Machinery (ACM) and Sigma Xi.