Virtual Landmarks for the Internet

Liying Tang
Department of Computer Science
Boston University
litang@cs.bu.edu

Mark Crovella
Department of Computer Science
Boston University
crovella@cs.bu.edu

ABSTRACT

Internet coordinate schemes have been proposed as a method for estimating minimum round trip time between hosts without direct measurement. In such a scheme, each host is assigned a set of coordinates, and Euclidean distance is used to form the desired estimate. Two key questions are: How accurate are coordinate schemes across the Internet as a whole? And: are coordinate assignment schemes fast enough, and scalable enough, for large scale use? In this paper we make contributions toward answering both those questions. Whereas the coordinate assignment problem has in the past been approached by nonlinear optimization, we develop a faster method based on dimensionality reduction of the Lipschitz embedding. We show that this method is reasonably accurate, even when applied to measurements spanning the Internet, and that it naturally leads to a scalable measurement strategy based on the notion of *virtual landmarks.*

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design; C.2.4 [**Computer-Communication Networks**]: Distributed Systems—*Distributed applications*

General Terms

Algorithms, Measurement, Performance

Keywords

Network Distance, Principal Component Analysis, Network Coordinates

1. INTRODUCTION

In many emerging applications there is opportunity for optimization based on knowledge of point-to-point delays in the underlying network. Example applications that can benefit from such knowledge include content delivery networks, peer-to-peer networks, multiuser games, overlay routing networks, and applications employing dynamic server selection. However, active measurement of network delays can be difficult, time consuming, and can add to network load. In response a number of methods have been proposed for estimation of network delays based on reduced or incomplete measurements [10, 9, 8, 18, 11].

A promising method proposed in [18] is to assign *coordinates* to nodes. The idea is to assign coordinates in such a manner that the associated Euclidean distance approximates network delay (round-trip propagation and transmission time). A node's coordinates are determined based on measurements to a fixed set of designated nodes, called landmarks.

This method has a number of attractive properties as compared to complete measurement of all inter-node delays. The primary attraction is that it allows approximating a large number of network distances (*i.e.*, network delays) using only a small number of actual measurements. That is, once two nodes have determined their coordinates, the network distance between them can be estimated without any further network measurement. Furthermore, the method allows distance between two nodes to be estimated by any third party that knows the nodes' coordinates. Finally, the use of Euclidean space to represent node location is attractive because of the large set of methods available for analysis of point sets in Euclidean space, including methods for search, partitioning, clustering, and dimensionality reduction.

The general problem of assigning coordinates in this manner is one of embedding a finite metric space into a Euclidean space [16]. In the case of network distances, an exact embedding is not expected to be possible, so one seeks an embedding that minimizes an error metric on distances. The resulting problem is a nonlinear optimization, which in work to date has been approached using iterative methods [18, 27].

The authors in [18] show that Euclidean embedding of network distances is reasonably accurate for the dataset they studied, which consists of about 16,500 node pairs. This is rather surprising because network delays are not *a priori* derived from measurements of a Euclidean space. In fact, network delays do not necessarily satisfy the triangle inequality, and so only form a semi-metric space. Thus a natural question is how accurate Euclidean embedding of network distances is over a wider range of datasets, and over datasets that span a larger fraction of the networks comprising the Internet.

This question is the first focus of this paper. We examine seven datasets, including two that attempt to capture measurements from a diverse set of destinations throughout the Internet. Across all of our datasets, we find that the accuracy of Euclidean embedding is remarkably good; typically approximately 90% of the distances have relative error less than 0.5. This encouraging result justifies further algorithmic exploration of Euclidean embeddings.

Assuming such embeddings can be accurate, two remaining bar-

IMC'03, October 27–29, 2003, Miami Beach, Florida, USA.

riers to their wide deployment are the computational complexity of coordinate determination, and the scalability of the associated measurement process. The first barrier arises because to date optimal network distance embeddings have been obtained using iterative algorithms for nonlinear optimization. This method is computationally expensive, which becomes an issue when working with large datasets, or if nodes must frequently recalculate their coordinates. Furthermore, this method is complex; it depends on proper settings of a number of parameters to guide the optimization process. The second barrier arises because each node determines its coordinates from measurements to a fixed set of landmarks; thus the measurement traffic arriving at the landmarks increases in proportion to the number of nodes in the system.

These problems are the second focus of this paper. To generate fast embeddings of network distances we employ the Lipschitz embedding [6]. The basic idea of the Lipschitz embedding is to use distances as coordinates. To find the coordinate vector $\vec{x}_i \in \mathbb{R}^n$ for node i, one sets the jth component of $\vec{x}_i$ to the measured distance between node i and landmark j, for $j = 1, ..., n$.

The Lipschitz embedding can be accurate because two objects that are close to each other in a metric space typically have similar distances to many other objects. Thus two nearby points in the original metric space may have very similar coordinate vectors, and so may map to nearby points under the Lipschitz embedding.

As we discuss in later sections, the accuracy of the Lipschitz embedding improves with the number of landmarks used. However we show that it is not necessary for the number of components in the coordinate vector to grow with the number of landmarks used. That is, even when using (say) 20 landmarks, it is possible to construct an accurate embedding using coordinates with only 7 to 9 components.

To do this we construct a low-dimensional embedding from a high dimensional Lipschitz embedding using principal component analysis (PCA). We start by taking measurements from m nodes to n landmarks (landmarks can be chosen at random). The result is an $m \times n$ matrix A in which row i is the initial n-dimensional coordinate vector $\vec{x}_i$ for node i. By applying principal component analysis to A, we can map each $\vec{x}_i$ to a new $\vec{y}_i$ in a lower dimensional space, while approximately preserving distances. That is we map $\vec{x}_i \in \mathbb{R}^n$ to $\vec{y}_i \in \mathbb{R}^r$ such that $r \ll n$ and $||\vec{x}_i - \vec{x}_j|| \approx ||\vec{y}_i - \vec{y}_j||$ for all $i, j \in 1, ..., m$.

The mapping from $\vec{x}_i$ to $\vec{y}_i$ obtained via PCA is a linear one. That is, $\vec{y}_i = M\vec{x}_i$ for some M (where M is an $r \times n$ matrix, and $\vec{x}_i$ and $\vec{y}_i$ are treated as column vectors). Thus we can think of the final coordinates of node i (the components of $\vec{y}_i$) as distances to *virtual landmarks*. The distance to a virtual landmark is defined as a linear combination of distances to actual landmarks.

Virtual landmarks provide significant advantages. We show that once M has been computed using some set of m nodes, it can be used to accurately compute coordinates for all nodes (not just those involved in coming up with M). This means that the construction of a node's coordinate vector from network distance measurements is a fast, simple *matrix-vector multiplication* (rather than an iterative nonlinear optimization with parameters that require tuning). Furthermore, the accuracy of the Lipschitz embedding can outperform that of the nonlinear optimization approach used in [18], although the relative performance of the two methods differs on different datasets. Lastly, we show that similar virtual landmarks can be constructed from different sets of actual landmarks. This leads to a natural approach to measurement scalability in which the same virtual landmarks are constructed by different nodes using measurements to distinct sets of actual landmarks.

The success of our method relies on empirically observed properties of network delay measurements, which we find to be consistent across all the datasets we examine. When one interprets inter-node measurements as coordinates, one can easily construct a very high-dimensional space — there are 116 dimensions in the case of one of our datasets. However, all of our datasets suggest that within this space, points tend to lie on a low dimensional Euclidean manifold (of dimension approximately 7) and that within this manifold, the desirable properties of the Lipschitz embedding approximately hold – namely, that distance within this manifold approximates actual network distance.

In the remainder of the paper we first summarize relevant background and related work (Section 2). We then describe the datasets we use in Section 3, and present their characteristics related to Euclidean embedding in Section 4. We then develop our embedding method in Section 5 and discuss its scalability properties in Section 6. We conclude in Section 7.

2. BACKGROUND AND RELATED WORK

2.1 The Embedding Problem

The embedding problem – the problem of inferring coordinates from distances – is an old one, spanning many fields, including mathematics and various branches of theoretical and applied computer science.

The embedding problem can be cast as follows. A *metric space* is a pair (X, d) where X is a set, and d is a *metric*. A metric is a distance function such that for $x, y, z \in X$, d satisfies 1) symmetry: $d(x, y) = d(y, x)$; 2) positive definiteness: $d(x, y) \geq 0$ with equality iff $x = y$; and 3) the triangle inequality: $d(x, y) + d(x, z) \geq d(y, z)$. A *Hilbert space* is a metric vector space, that is, a metric space in which $X = \mathbb{R}^n$ for some n. A *Euclidean space* is a Hilbert space in which the metric is the Euclidean or l_2 norm $d(\vec{x}, \vec{y}) = ||\vec{x} - \vec{y}||_2$.

An *embedding* of a metric space (X, d) into a Hilbert space $(\mathbb{R}^n, \delta)$ is a mapping $\phi : X \to \mathbb{R}^n$. In this paper (X, d) will always be a finite metric space (*i.e.,* X is a finite set) and $(\mathbb{R}^n, \delta)$ will always be a Euclidean space. The *distortion* of an embedding is the smallest value of c_2/c_1 that guarantees that

$$c_1 \cdot d(x, y) \leq \delta(\phi(x), \phi(y)) \leq c_2 \cdot d(x, y)$$

for all pairs $x, y \in X$ and $c_1, c_2 > 0$. An embedding with distortion 1 is an *isometry.* Distortion is a worst-case measure of the quality of an embedding; an average-case measure is *stress.* The stress of an embedding is

$$\frac{\sum_{x,y \in X} (\delta(\phi(x), \phi(y)) - d(x, y))^2}{\sum_{x,y \in X} d(x, y)^2}.$$

Given (X, d) and $(\mathbb{R}^n, \delta)$, the *embedding problem* consists of finding a ϕ that minimizes distortion, stress, or a similar error metric on distances. In general, finding an exact solution is a nonlinear optimization problem, because in Euclidean space, δ is a nonlinear function of its arguments.

In the special case where an isometry exists, it can be found via the methods of *distance geometry* [5]. Methods based on distance geometry generally assume complete, exact distance measurements, and so must be extended to handle cases when data is incomplete, or points are not exactly embeddable; hence it is best suited for problems (such as determination of molecular structure from NMR data [30]) in which the distance measurements are actually drawn from an underlying Euclidean space of known dimension. Our investigations suggest that distance geometric approaches are not sufficiently robust in the face of the significant

distortion required for a Euclidean embedding of network distance. The approaches we present in this paper significantly outperformed distance geometric methods on the datasets we studied.

The most commonly used term for the process of embedding an arbitrary finite metric space into a Euclidean space is *multidimensional scaling (MDS)* [31]. Applications of distance geometry to the embedding problem are often referred to as *classical metric* MDS [28]. The general term MDS usually refers to incremental methods (such as steepest descent) applied to the minimization of stress. A good review of commonly used embedding methods is [12].

In contrast to MDS, a different way of finding a good ϕ is the *Lipschitz embedding.* A Lipschitz embedding is defined in terms of a set D of subsets of X, $D = \{L_1, L_2, ..., L_n\}$. We extend the distance function d to a set $L \subset X$ as follows: let $d(x, L) = \min_{y \in L} d(x, y)$. Then the Lipschitz embedding of X with respect to D is the mapping ϕ such that

$$\phi(x) = [d(x, L_1), d(x, L_2), ..., d(x, L_n)]$$

where $|D| = n$ and the mapping is into the vector space $\mathbb{R}^n$. Thus, the Lipschitz embedding defines a coordinate space in which each axis i corresponds to the minimum distance from x to the nearest member of L_i. In this paper we only consider Lipschitz embeddings in which each L_i is a singleton, and we call the various L_is *landmarks.* Hence the interpretation of our embedding is particularly simple: component i of the vector $\phi(x)$ is simply the distance from x to landmark i.

The intuition behind the Lipschitz embedding is that in a metric space, two nearby points (say, a and b) will typically have similar distances to a third point (say, x). This is required by the triangle inequality:

$$|d(a, x) - d(b, x)| \leq d(a, b).$$

So under a Lipschitz embedding ϕ, the magnitude of the difference between any component of $\phi(a)$ and the corresponding component of $\phi(b)$ is bounded above by $d(a, b)$.

It is important to note that although two nearby points a and b will have similar Lipschitz coordinate vectors, the magnitude of the distance between $\phi(a)$ and $\phi(b)$ is affected by the number of dimensions used in the embedding. For this reason it is necessary to normalize the embedding by estimating the ratio β between distances in the Lipschitz embedding and the original metric space. This is generally done by least squares minimization:

$$\beta = \arg\min_{\theta} \sum_{x,y \in X} (\theta \, ||\phi(x) - \phi(y)||_2 - d(x, y))^2.$$

Thus, in the remainder of the paper, when we refer to the Lipschitz embedding of a node x we mean $\beta \cdot \phi(x)$, where β is computed relative to all nodes in the given dataset.

The Lipschitz embedding is a powerful method. It is the basis for a number of important theoretical results on minimum-distortion embeddings [13, 6, 17]. Furthermore, it has been used in other settings for related problems. In [4, 19] the authors used Lipschitz embeddings to find nearby nodes for the purpose of geolocation. Furthemore, [19] uses delay as the basis for constructing its Lipschitz embedding of Internet hosts, as in this paper, and uses the Euclidean distance to known locations in the embedding as an indicator of geographic proximity to those locations. However, in those studies, the Lipschitz embedding was an intermediate step in addressing geolocation; our paper does not address geolocation, but rather focuses on the properties of the Lipschitz embedding itself.

It is clear that the quality of the results obtained via that Lipschitz embedding depends on the choice of landmarks [29, 3]. In Section 5 we show that quality generally improves as the number of landmarks increases. Thus, to create low-dimensional embeddings, we employ dimensionality reduction based on principal component analysis (described in Section 2.3), leading to the notion of virtual landmarks, described in detail in Section 5.

2.2 Embedding Network Distances

As described in Section 1, the first paper to explore Euclidean embedding of network distances was [18]. The metric of interest is minimum round trip time – specifically, the minimum over a set of measured delays of ICMP echo messages. Minimum RTT measures the round trip time of a packet in the absence of congestion; this value is assumed to change infrequently enough that it is a worthwhile target for efficient measurement methods. We make the same assumption in this paper (however we believe that our methods have potential to be extended and applied to more dynamic network measures).

The embedding method used in [18], called Global Network Positioning (GNP), works as follows: First, a set of landmarks each measure their distances to all other landmarks, and calculate their coordinates using MDS (iterative nonlinear optimization of the stress metric). Next, each node in the system measures distances to each landmark, and then performs its own nonlinear optimization to compute its coordinates.

The authors in [27] propose another approach to embedding network distances which compares favorably in accuracy to GNP on synthetically generated graphs. However the method is also an iterative nonlinear optimization, with parameters that must be tuned, and is considerably more complex than GNP.

In contrast to these approaches, the methods we propose in this paper are much simpler, and more computationally efficient. Furthermore, the accuracy of our method is comparable to GNP for the datasets we examine.

The methods proposed in [20] are focused on enhancing scalability of coordinate schemes; as in our work, they propose the use of multiple landmark sets to distribute the load due to measurement traffic. However their methods are based on a distance geometric embedding, which we have found lacks robustness across diverse datasets.

In work performed concurrently with that in this paper, the authors in [15] propose methods very similar to ours. While [15] focuses on comparison to a wider set of alternative distance estimation methods, this paper concentrates to a greater degree on examining the method's performance on a wider variety of Internet measurements. Furthermore, the two papers take different approaches to improving the scalability of the basic scheme.

Finally, work that provided some of the inspiration for our paper is described in [7]. That paper is concerned with clustering the IP address space for efficient route selection, and implicitly uses Euclidean distance in a Lipschitz embedding as the clustering metric. A similar clustering method is proposed in [8].

2.3 Principal Component Analysis

Portions of our analysis and approach rely on principal component analysis (PCA, also called the Karhunen-Lòeve Transform). Here we briefly review the relevant aspects of PCA. For more details, see [25, 21].

PCA is based on singular value decomposition (SVD), which is a matrix factorization with a number of advantageous properties. SVD factors an $m \times n$ matrix A into the product of three matrices U, W, and V^T. U is $m \times n$, and W and V are both $n \times n$. That

is,

$$A = U \cdot W \cdot V^T \quad (1)$$

This factorization has three key properties:

1. V is orthogonal and U is column-orthogonal. Hence, multiplication of a vector by V does not change its length, and multiplication of a set of vectors by V does not change the distances between them. (All vectors here and below are column vectors unless otherwise specified. Furthermore, we will use the terms *vector* and *point* interchangeably.)

2. W is a diagonal matrix with nonnegative entries w_i on the diagonal. We assume that the entries are ordered in decreasing value (which can be obtained via a corresponding reordering of the columns of U and rows of V^T). The nonzero diagonal entries w_i are called the *singular values* of A.

3. The columns of U are eigenvectors of AA^T and the rows of V^T are eigenvectors of A^TA. The values of w_i that are nonzero are the square roots of the nonzero eigenvalues of both AA^T and A^TA.

Note that (1) can be rewritten to express matrix A as a sum of outer products of columns of U and rows of V^T, with the weighting factors being the singular values w_k:

$$A_{ij} = \sum_{k=1}^{n} w_k U_{ik} V_{jk} \quad 1 \le i \le m, 1 \le j \le n$$

The central idea of principal component analysis (PCA) is that often we can approximate A by only a few terms in the sum, in which case only a subset of the columns of U and of the rows of V^T need to be stored.

In fact, SVD yields the *optimal* linear approximations to A in a mean-square sense. In particular, let $v_{(i)}$ be the ith row of V^T; it can be shown that $v_{(1)}$ points in the direction of *maximum variance* in the point set corresponding to the rows of A. Furthermore, each subsequent row of V captures the maximum variance possible among the remaining orthogonal directions:

$$v_{(n)} = \arg \max_{||x||=1} ||(A - \sum_{i=1}^{n-1} w_i' v_{(i)}^T)x||$$

where w_i' is a column vector of all w_i's. The vector $v_{(i)}$ is referred to as the i^{th} principal component of the point set represented by the rows of A.

Thus, when we approximate the point set A by a partial sum

$$A_{ij}' = \sum_{k=1}^{r} w_k U_{ik} V_{jk} \quad 1 \le i \le m, 1 \le j \le n$$

with $r < n$, we obtain a new set of points in $\mathbb{R}^r$ that is the "best" linear transformation of those points from the higher dimensional space $\mathbb{R}^n$ to the lower dimensional space $\mathbb{R}^r$. This is illustrated in Figure 1. The figure shows a set of points in two dimensions, along with the first principal component, representing the optimal linear mapping (in a mean-square sense) of those points to a single dimension.

Thus PCA is the application of SVD to dimensionality reduction. Since each column of U and each row of V^T is a unit vector, the magnitude of the contribution of dimension i to overall variance of the point set A' is entirely determined by the value w_i. Thus, if the first r singular values are much bigger than the rest, we can expect to introduce very little error by reducing from n to r dimensions.

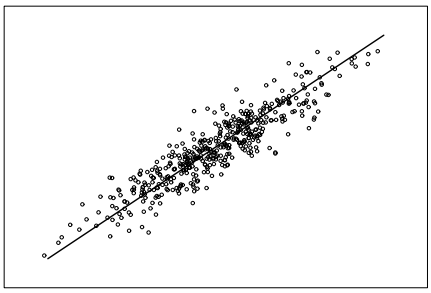

Figure 1: Principal Component Analysis applied to a 2-dimensional dataset

This suggests that an important way to assess the potential for dimensionality reduction in an empirical dataset is to plot its singular values in decreasing order. Such a plot is called a *scree plot*; if the scree plot shows a knee after some number of values r, this is good evidence that r dimensions are sufficient to capture most of the variability in the dataset.

3. DATASETS STUDIED

As we will show in Sections 4 and 5, the properties of network delays as actually measured in the Internet are crucial to the success of the methods we describe in this paper. While previous studies of Euclidean embedding have looked at only one or two datasets of moderate size (tens of thousands of measurements), we base our results on seven datasets collected at different times and locations, and including two large datasets (each comprising millions of measurements).

We will often cast a dataset as a matrix A. A will always be $m \times n$, with $m \ge n$. The nodes corresponding to rows are $\mathcal{R} = \{\mathcal{R}_i,\ i = 1, ..., m\}$, and the nodes corresponding to columns are $\mathcal{C} = \{\mathcal{C}_i,\ i = 1, ..., n\}$. Then $A_{ij} = d(\mathcal{R}_i, \mathcal{C}_j)$. When $m = n$ and $\mathcal{R}_i = \mathcal{C}_i$ for all $i = 1, ..., n$, the matrix (or dataset) is *symmetric*; otherwise it is asymmetric. For a symmetric dataset, will use $\mathcal{N}$ to refer to either $\mathcal{R}$ or $\mathcal{C}$ since they are identical in that case.

The methods we use assume no missing entries in the matrix A. However, in practice for large datasets, inevitably some entries will be missing. We removed the rows containing missing entries in the two large datasets, which comprised less than 2% of the data in both cases.

3.1 GNP

The GNP project [18] measured minimum round trip time between 19 active sites and 869 targets. In addition, measurements were taken between all pairs of active sites. These are respectively referred to as the GNP asymmetric and GNP symmetric datasets, and were collected in May 2001. About half of the targets, and 12 of the probes are in North America; the rest are distributed globally. Target IP addresses were chosen by probing the IP address space randomly. Distance between two hosts was taken as the minimum time among 220 pings sent one second apart.

3.2 RON

The RON project [2] collected a variety of measurements including ping latency between all pairs of nodes in an overlay network. Two datasets are available, both symmetric: RON1 is taken from a 13 $\times$ 13 mesh, and RON2 is taken from a 15 $\times$ 15 mesh; these

were collected in March 2001 and May 2001, respectively. The vast majority of RON sites are in the US. Network distance in each case is the minimum of thousands of ping times.

3.3 NLANR AMP

The NLANR Active Measurement Project [1] collects a variety of measurements between all pairs of participating nodes. Most participating nodes are at NSF supported HPC sites, and so have direct connections to the Abilene network; about 10% are outside the US. The dataset we use was collected on January 30, 2003 and is symmetric, consisting of measurements of a 116 $\times$ 116 mesh. Each host was pinged once per minute, and network distance is taken as the minimum of the ping times over the day's worth of data.

3.4 Skitter

The Skitter project [23] is a large-scale effort to continuously monitor routing and connectivity across the Internet. It consists of approximately 19 active sites that send probes to hundreds of thousands of targets, including the set of active sites. Thus this dataset has a symmetric subset, but is primarily asymmetric. Targets are chosen so as to sample a wide range of prefixes spanning the Internet. Results reported in [14] suggest that about 50% of the targets in this dataset are outside the US. The dataset we used was collected in December 2002; each target was pinged approximately once per day. Network distances are the minimum ping time over 12 days. The set of targets varies among active sites; selecting the largest set of rows for which complete data is available yields a 12 $\times$ 12 symmetric dataset, and a 12 $\times$ 196,286 asymmetric dataset with 2,355,565 entries.

3.5 Sockeye

Our last dataset was collected at Sockeye Networks [24]. It consists of measurements from 11 active sites to 156,359 locations. Targets were chosen through a scheme designed to efficiently explore as much of the routable Internet as possible. From a composite BGP table obtained from RouteViews [22], all listed /24 prefixes were extracted (over 4,600,000 prefixes). Addresses within each prefix were probed, seeking live addresses; approximately 745,000 prefixes were found to have live addresses. A traceroute was performed to a live address in each such prefix, and the penultimate hop was identified. The resulting set of 158,707 penultimate hops comprised the targets used. Removing rows with missing measurements resulted in a 11 $\times$ 156,359 asymmetric dataset with 1,719,949 entries. Each active site sent a ping to each target on an hourly basis. Network distance was taken as the minimum of all pings over a single day.

4. CHARACTERIZING NETWORK DISTANCE

Before we describe our method for Euclidean embedding of network distance measurements, it is helpful to examine some properties of our network distance datasets. The properties we describe shed light on why our Euclidean embedding method works well.

4.1 Violations of the Triangle Inequality

The first question we address concerns the assumption implicit in the Lipschitz embedding, namely, that network distances obey the triangle inequality. It is clear that Internet traffic does not always follow the shortest possible path [26] and that there is potential for violation of the triangle inequality due to routing policy. Thus, before exploring the Lipschitz embedding it is helpful to quantify, in frequency as well as severity, violations of the triangle inequality in network distance.

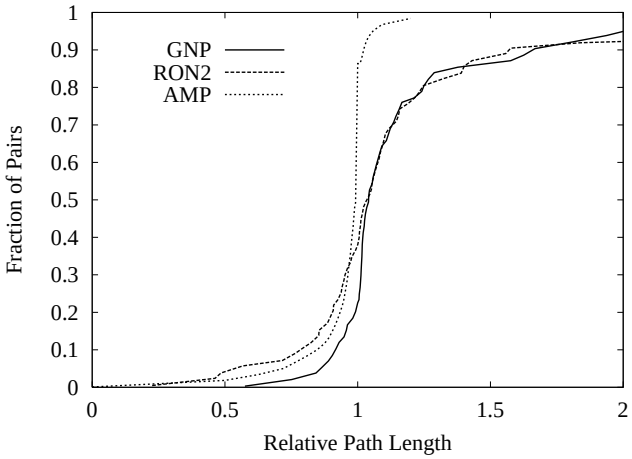

Figure 2: Severity of triangle inequality violations: the cumulative distribution of $\min_k(d(i,k)+d(k,j))/d(i,j)$ over all pairs i, j.

To measure the frequency of triangle inequality violations, we ask how many two-hop paths are shorter than the corresponding one-hop path. That is, for all combinations of i, j, k, we count how many times $d(i,k) + d(k,j) < d(i,j)$. Our results show that for the GNP symmetric dataset, this occurs 3.7% of the time; for RON2, 6.3% of the time; and for AMP, 1.4% of the time. Thus triangle inequality violations are not particularly frequent.

We measure the severity of triangle inequality violations as follows: for each node pair, we ask what the shortest path is between the two that passes through a third node. That is, for all pairs of nodes i and j, we compare $d(i,j)$ with $\min_k(d(i,k) + d(k,j))$. The relative change in path length passing through the best alternate path is $\min_k(d(i,k) + d(k,j))/d(i,j)$. The cumulative distribution of this quantity is presented for three symmetric datasets in Figure 2.

The figure shows that in fact many node pairs participate in non-triangular combinations with a third node. As many as 40% of the node pairs have a shorter path through an alternate node. However the severity of violations is not particularly great; for only 10% of the node pairs is there an alternate path that is more than 20% shorter.

4.2 Intrinsic Dimensionality

The next question concerns the best number of dimensions to use in embedding network distances in a Euclidean space. While increasing the number of dimensions will always increase the accuracy of the embedding, we seek to find a small number of dimensions that captures most of the information in our datasets.

To assess the appropriate number of dimensions needed for embedding, we examine scree plots of the distance matrix A. Then the scree plot of A can give some insight into the intrinsic dimensionality of the underlying point set. To demonstrate this we perform the following experiment. We start with 250 points distributed uniformly at random in a unit hypercube of dimension n. We then form the 250 $\times$ 250 distance matrix A and examine its scree plot. The result is shown in Figure 3 for $n = 5$, 10, 15 and 20.

The figure shows that when the points are distributed in n-dimensional space, the first n singular values of the distance matrix are significantly larger than the rest. This is evidenced by the "knee" in the scree plot at the corresponding value of n. This simple experiment suggests that we can gain some insight into the in-

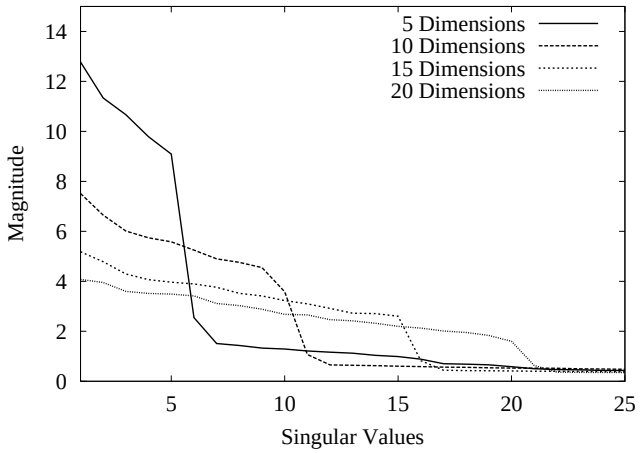

Figure 3: Scree plots of distance matrices for synthetic point sets in varying numbers of dimensions.

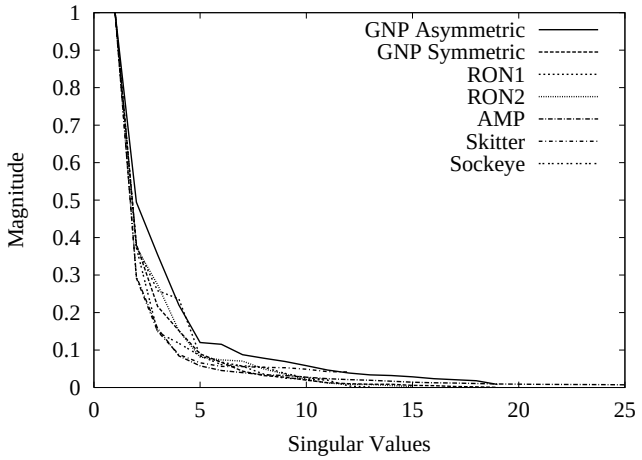

Figure 4: Scree plots of empirical distance matrices.

trinsic dimensionality—the number of dimensions necessary to accurately embed a set of distances—by examining the scree plot of the distance matrix.

In Figure 4 we show scree plots for all our datasets (for ease of comparison the largest singular value in each case is normalized to 1). This figure shows that these datasets have two surprising features:

1. All of the datasets have remarkably similar sets of singular values. There is not much difference between the distribution of singular values for the small, geographically localized datasets (RON1, RON2), the medium sized datasets (GNP, AMP), and the large, geographically dispersed datasets (Skitter, Sockeye).

2. The number of large singular values in each dataset is only about 7 to 9, even though most matrices have many more columns (and one – AMP – has 116 columns).

Of course, our empirical distance matrices are not taken from a Euclidean space, so comparing Figures 3 and 4 is only suggestive. However the scree plots provide some justification for the observation made in [18] that approximately 7 dimensions is sufficient for accurate embedding of the GNP dataset.

Dataset	Mean Rel. Error
GNP Symmetric	8%
GNP Asymmetric	25%
RON1	10%
RON2	8%
AMP	12%
Skitter	10%
Sockeye	13%

Table 1: Mean relative error when compressing datasets to 7 dimensions.

Furthermore, as discussed in Section 2.3, these results suggest that each distance matrix can be compressed without much loss of accuracy as follows. Using SVD, factor A into $U \cdot W \cdot V^T$. Now let U' be the r leftmost columns of U, let W' be the upper left $r \times r$ submatrix of W, and let V'^T be the r uppermost rows of V^T. Then we only need to store U', W', and V'^T in order to reconstruct a good approximation to A as:

$$A' = U' \cdot W' \cdot V'^T. \quad (2)$$

To illustrate this, we compress each dataset to r = 7 dimensions. This results in compression factors from about 2 in the case of Sockeye to about 8 for AMP. We then measure the mean relative error between the entries in the resulting A' and those in the original matrix of measured data A. The results are shown in Table 1. The Table shows that preserving only 7 columns of each of the factors U and V is sufficient to capture almost all of the information in any of the network distance datasets we examined.

5. LIPSCHITZ COORDINATES

The previous section showed that our datasets appear to be suitable for Lipschitz embedding: the triangle inequality usually holds, and a small number of dimensions (perhaps 7) should be sufficient.

Applying the Lipschitz embedding to our datasets is straightforward. Consider a distance matrix A. Then any subset of the columns of A defines a Lipschitz embedding of $\mathcal{R}$, and any subset of the rows of A defines a Lipschitz embedding of $\mathcal{C}$.

For example, assume we seek an embedding for a set of nodes with symmetric distance matrix A. Then we seek to find a ϕ that embeds the finite metric space $(\mathcal{N}, d)$ into the Euclidean space $(\mathbb{R}^r, \delta)$, where $d(\mathcal{N}_i, \mathcal{N}_j) = A_{ij}$ and δ is the Euclidean norm. The Lipschitz embedding consists of choosing r columns of A, to form an $m \times r$ matrix which we denote L. Then $\phi(\mathcal{N}_i) = L_{(i)}$, where $L_{(i)}$ is row i of L interpreted as a vector in $\mathbb{R}^r$.

However, the choice of how many rows or columns to use in forming L has a strong effect on the quality of the resulting embedding. This is the problem of landmark selection, introduced in Section 2.1. We illustrate the problem of landmark selection via an example with synthetic data, as follows.

We distribute 250 points uniformly at random in a 20-dimensional unit hypercube, and calculate the resulting distance matrix A. We then study the quality of the Lipschitz embedding of A as a function of the landmarks used. In the first case we use as landmarks points that lie at location 10 on each of the axes (*i.e.,* for each point, 19 coordinates are zero and one has value 10). These points are not part of the dataset, and are very far from any points in the dataset. Although unrealistic, they provide a near-optimal comparison. In the other cases we use 20, 50, or 100 randomly chosen landmarks.

The results are shown in Figure 5, which plots the cumulative

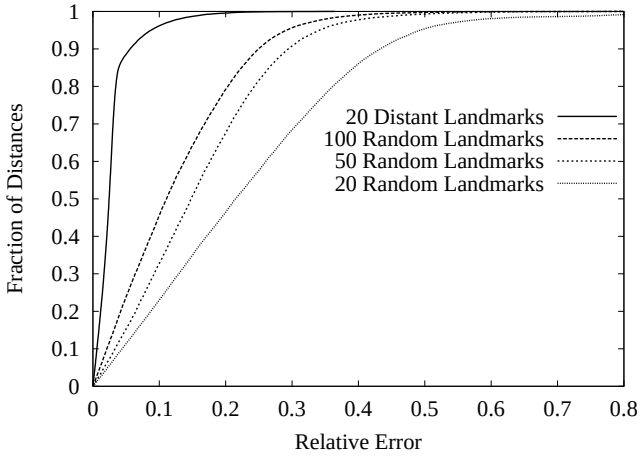

Figure 5: Effect of different landmark sets on accuracy of Lipschitz embedding for synthetic data. Cumulative distribution of relative error of embedding.

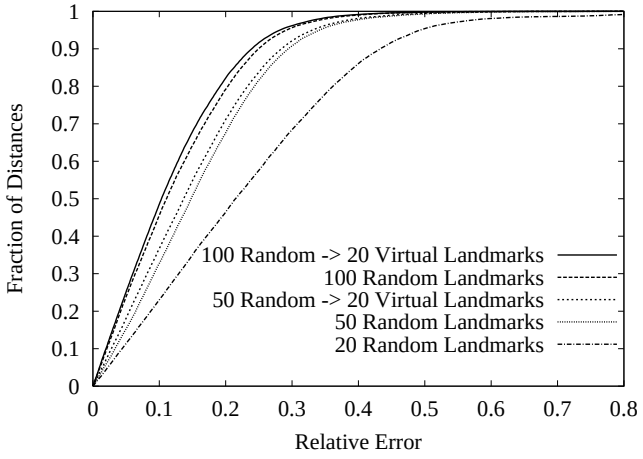

Figure 6: Effect of applying PCA to landmark set for synthetic data. Cumulative distribution of relative error of embedding.

distribution of the relative error of each Lipschitz embedding across all 62,500 distance measurements. The results are in agreement with intuition. The choice of 20 random landmarks is the worst one, since at least 20 points are needed to span the underlying space. As we increase the number of landmarks, the accuracy improves toward the optimal (20 distant landmarks).

Unfortunately, using 100 landmarks implies assigning a coordinate vector of length 100, which is undesirable. To address this problem we make use of the observations made in Section 4. That section showed that the column space of A is typically captured in roughly 7 to 9 dimensions. Furthermore, we note that approximating L with L' by applying PCA via (2) does not significantly alter the *distances* between the points corresponding to the rows of L and L'. So we can form a new embedding of the points corresponding to an embedding L by computing LV'. Since $L \approx L'$, $LV' \approx U'W'$.

Thus LV' is an $m \times r$ matrix in which the rows are a *new* embedding of $\mathcal{R}$. In similar fashion, we can calculate the new embedding

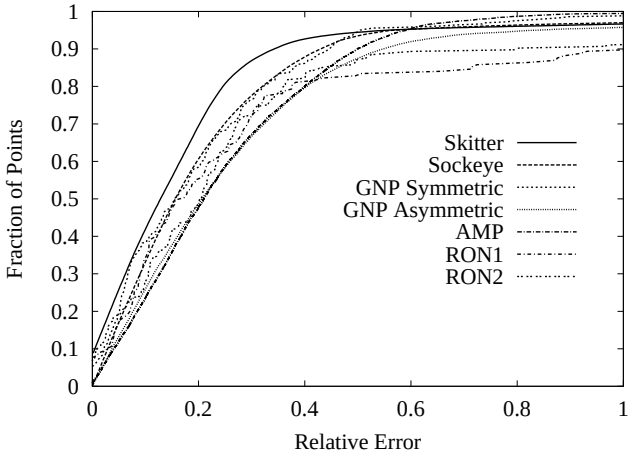

Figure 7: Quality of the embedding by virtual landmarks.

$\vec{y} \in \mathbb{R}^r$ for any point whose Lipschitz embedding is $\vec{x} \in \mathbb{R}^n$ as:

$$\vec{y} = V'^T \vec{x}.$$

Note that the components of $\vec{y}$ are linear combinations of the components of $\vec{x}$, which are themselves distances. So a component of $\vec{y}$ is a linear combinations of distances, and we can think of it as a distance to a *virtual* landmark.

We first illustrate this idea using our synthetic data. For each of the two $250 \times n$ matrices corresponding to the Lipschitz embeddings that use $n = 50$ or 100 random landmarks, we apply PCA and reduce it to a 250×20 matrix. The rows of this smaller matrix are an embedding using 20 virtual landmarks. The resulting relative error distribution is shown in Figure 6. Note that in each case, not only is using the 20 virtual landmarks much more accurate than using 20 actual landmarks, it is at least as accurate as using the actual landmarks (50 or 100) that were used as input to the method.

Turning now to empirical measurements, we applied the method of virtual landmarks to our datasets. For each dataset, we performed the virtual landmarks embedding using 8 dimensions.[1] The results are shown in Figure 7 as cumulative distribution plots of relative error.

The figure shows that the method results in embeddings that are accurate, and that accuracy is remarkably consistent across datasets. Furthermore, the best performance is achieved on the largest datasets (Sockeye and Skitter). For Skitter, 90% of the distances have relative error less than 0.34. In the other cases embedding by virtual landmarks is sufficient to recreate 90% of the distance measurements with relative error less than 0.5.[2] This level of performance is comparable to or better than that reported in [18] for the GNP method.

We compare the accuracy of the virtual landmarks approach to the GNP embedding method in Figures 8 and 9. Figure 8 compares the methods using the AMP dataset. This plot shows that the method of virtual landmarks performs much better than the GNP method for this dataset. Furthermore, the difference in computational requirements is large: running on a Sun Ultra E450, the

[1] In these experiments all rows of A were used in the PCA step; we leave for future work the question of the smallest number of rows useful for reasonable accuracy, and the proper choice of those rows.

[2] Since measurements between landmarks were not available for the Skitter and Sockeye datasets, coordinates for the landmarks themselves cannot be computed. As a result we estimated landmark coordinates using those of nearby nodes for those datasets.

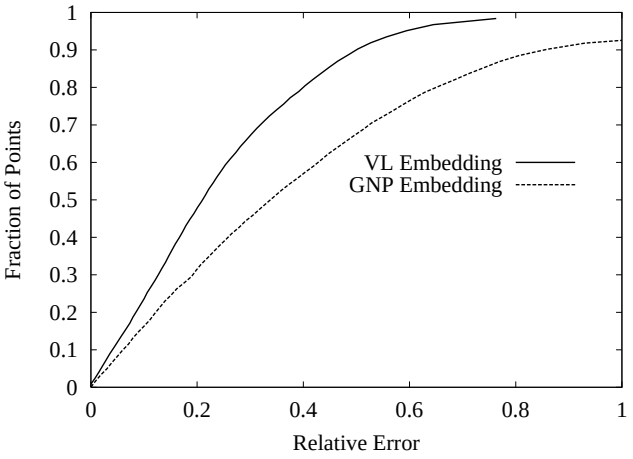

Figure 8: Comparison of virtual landmarks embedding with GNP embedding on AMP dataset. The same set of 20 landmarks was used in each case, yielding an embedding in 8 dimensions. The virtual landmark method required less than 1 second of computation, while the GNP method required over 1 hour.

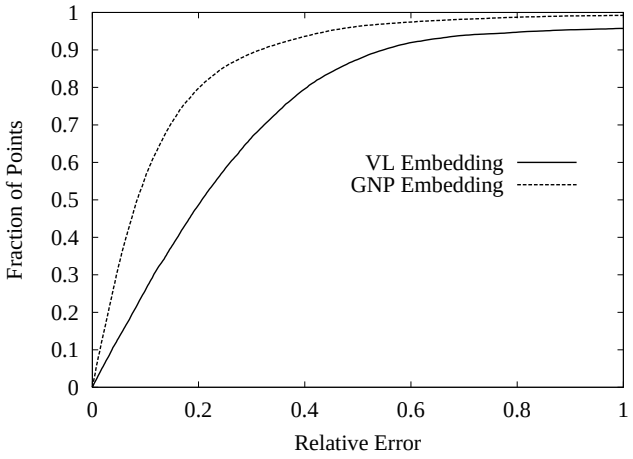

Figure 9: Comparison of virtual landmarks embedding with GNP embedding on GNP Asymmetric dataset. The same set of 20 landmarks was used in each case, yielding an embedding in 8 dimensions. The virtual landmark method required less than 1 second of computation, while the GNP method required over three minutes.

GNP method required 3,626 seconds of processing time, while the virtual landmarks method required less than 1 second.

Figure 9 compares the methods using the GNP dataset. Here we see that the virtual landmarks method performs worse than the GNP method, although still reasonably well (88% of distances have relative error less than 0.5). The difference in computational requirements is still significant: 182 seconds for GNP compared to less than 1 second for the virtual landmarks method.

Unfortunately, the computational demands of GNP make it impossible to compare against the datasets where the virtual landmarks method performs best (Sockeye and Skitter). Computing embedding via virtual landmarks for those datasets took less than 15 seconds in each case.

These results suggest that in some settings, the virtual landmarks method can perform as well as the GNP method while requiring much less computation; however we also conclude that further exploration is needed to understand the reasons why the relative performance of the two methods differs in different settings.

6. SCALING THE METHOD

A significant drawback of all methods that use a fixed set of landmarks is that measurement traffic to the landmarks increases in proportion to the number of nodes participating in the system. This is true as well for the method as described in the previous section.

However, we show in this section that the notion of virtual landmarks makes it easy to distribute measurement load more evenly throughout the system. The basic insight is that, using the virtual landmarks method, a randomly chosen set of landmarks defines an embedding space that can be easily (linearly) mapped into the another embedding space derived from a different set of landmarks.

That is, let L_1 and L_2 be two different Lipschitz embeddings of a set of nodes $\mathcal{R}$. We can think of these embeddings as each being defined by selecting n columns from a distance matrix A. For each embedding, we can obtain a virtual landmarks embedding using PCA, as described in the previous section, yielding L_1V_1' and L_2V_2'. Since these two embeddings are both likely to be projections onto the first r principal components of A, it is likely that the two point sets are nearly isometrically embeddable. To find the isometry we can select r points and determine the transformation of their coordinates from L_2V_2' to L_1V_1'.

Let P_1 be the coordinates of r points from L_1V_1', and P_2 be the coordinates of the same r points in L_2V_2'. We arbitrarily choose the coordinate space of L_1V_1' to be canonical. Then we solve $P_2T_2 = P_1$ to find T_2. To increase accuracy in the presence of measurement error, we can use more than r points, in which case we solve for T_2 using least squares. Once we have T_2, then we can calculate the canonical coordinates of a node whose measurements are to landmarks in set 2 as:

$$\vec{y}_i = T_2^T {V'}_2^T \vec{x}_i \tag{3}$$

where $\vec{x}_i$ is the vector of measurements to landmarks in set 2.

This suggests the measurement scheme shown in Figure 10. Nodes needing coordinates are divided into multiple sets (represent by hosts "A" and "B" in the diagram). Each node in set i is equipped with its local V_i and T_i. A node performs measurements to landmark set i and calculates its coordinates using (3).

The transformation matrix T_i is calculated by a special set of nodes ("Spanners" in the diagram). Spanners measure coordinates to two landmark sets: set i, and some other set j for which ${V'}_j^T$ and T_j are already known. The spanners then solve $L_i{V'}_i^T T_i = L_j{V'}_j^T T_j$ for T_i.

We can assess the accuracy of this method using our AMP dataset. We select two landmark sets at random (each of size 20), and find the locations of all 76 remaining nodes in the canonical coordinate scheme two ways: directly via landmark set 1, and indirectly via landmark set 2 using the remaining nodes as spanners. We then measure the relative error in position for each point. The results of five replications of this process, shown as cumulative distributions in Figure 11, suggests that there is very little error introduced by calculating coordinates indirectly via spanners, rather than directly to the canonical landmark set.

7. CONCLUSIONS AND FUTURE WORK

In this paper we've explored the properties of network distance

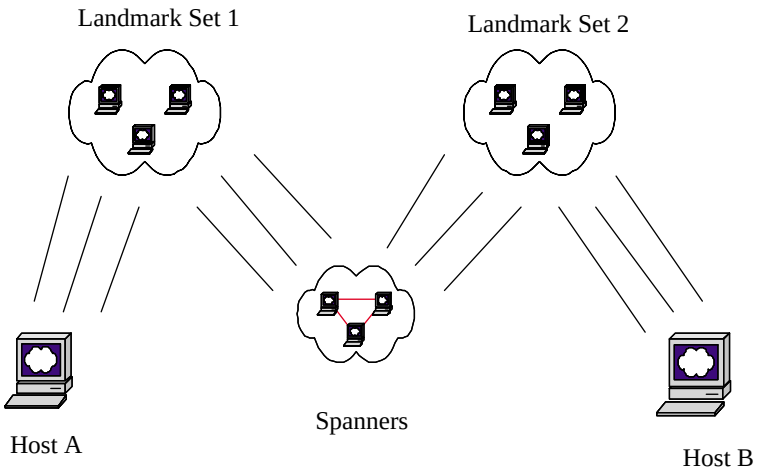

Figure 10: Diagram of Scaling Method

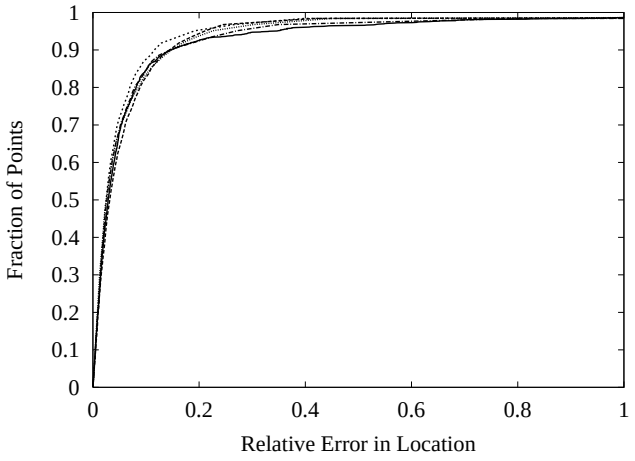

Figure 11: Accuracy of the Scaling Method

in the Internet. We've found, rather surprisingly, the network distances can generally be described as the linear combination of a small number of orthogonal vectors – typically 7 to 9. This suggests that an embedding in a Euclidean space of 7 to 9 dimensions is likely to be sufficient for reasonable accuracy.

We have also proposed and explored one such embedding, the Lipschitz embedding. By applying dimensionality reduction to the Lipschitz embedding, we obtain an embedding that has approximately the same accuracy with many fewer dimensions. The new dimensions are linear combinations of distances to landmarks, so we refer to them as distances to virtual landmarks.

We show that the method of virtual landmarks is simpler, and much faster, than previously proposed nonlinear optimization methods for embedding. Furthermore, the virtual landmarks method shows reasonable accuracy across a large variety of datasets. The method is more accurate than previously proposed methods on one dataset, and less accurate on another, which suggests that more investigation is warranted into the relative strengths of the method compared to nonlinear optimization. However, the overall performance of the method is good: approximately 90% of distances are preserved with relative error less than 0.5 on of our datasets, with the largest datasets showing even better performance.

Finally, we have shown that the virtual landmarks method is easily extended to a scalable measurement framework, that places only a constant measurement load on any host as the system size scales.

Acknowledgements

The authors wish to express their appreciation for very helpful conversations with Vassilis Athitsos, George Kollios, and Shang-Hua Teng at Boston University; Bruce Hendrickson at Sandia National Laboratories; and Geoffrey Brown at Sockeye Networks.

The Sockeye data used in this paper was collected at Sockeye Networks, `www.sockeye.com`. The RON data was collected by David Andersen at MIT, and the GNP data was collected by Eugene Ng at CMU. The NLANR AMP data used in this paper was collected by the National Laboratory for Applied Network Research with funding under National Science Foundation Cooperative Agreement ANI-9807479. The Skitter data was collected as part of CAIDA's Skitter initiative, `http://www.caida.org`. Support for Skitter is provided by DARPA, NSF, and CAIDA membership. The authors wish to express their appreciation to all these people and organizations for their data collection and distribution efforts.

This work was partially supported by NSF grants ANI-9986397 and ANI-0095988.

8. REFERENCES

[1] The NLANR active measurement project. http://amp.nlanr.net/active/.

[2] D. G. Andersen, H. Balakrishnan, M. F. Kaashoek, and R. Morris. Resilient overlay networks. In *Proceedings of the 18th ACM SOSP*, October 2001.

[3] V. Athitsos and S. Sclaroff. Estimating 3D hand pose from a cluttered image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, June 2003.

[4] P. Bahl and V. N. Padmanabhan. RADAR: An in-building RF-based user location and tracking system. In *Proceedings of IEEE Infocom*, pages 775–784, 2000.

[5] L. M. Blumenthal. *Theory and applications of distance geometry*. Chelsea Pub. Co., Bronx, N.Y., second edition, 1970.

[6] J. Bourgain. On Lipschitz embedding of finite metric spaces in Hilbert space. *Israel J. Math.*, 52(1–2):46–52, 1985.

[7] G. Brown. Internet address space clustering for intelligent route control. Submitted for publication, 2003.

[8] Y. Chen, K. Lim, R. H. Katz, and C. Overton. On the stability of network distance estimation. In *Proceedings of ACM SIGMETRICS Practical Aspects of Performance Analysis Workshop (PAPA 2002), in ACM SIGMETRICS Performance Evaluation Review*, September 2002.

[9] 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*, 9(5):525–540, 2001.

[10] P. Francis, S. Jamin, V. Paxson, L. Zhang, D. F. Gryniewicz, and Y. Jin. An architecture for a global Internet host distance estimation service. In *IEEE INFOCOM 1999*, pages 210–217, New York, NY, March 1999. IEEE.

[11] K. P. Gummadi, S. Saroiu, and S. D. Gribble. King: Estimating latency between arbitrary Internet end hosts. In *Proceedings of 2nd ACM Sigcomm Internet Measurement Workshop 2002*, November 2002.

[12] G. R. Hjaltason and H. Samet. Properties of embedding methods for similarity searching in metric spaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(5):530–549, May 2003.

[13] W. B. Johnson and J. Lindenstrauss. Extensions of Lipschitz mappings into a Hilbert space. In *Conference in modern analysis and probability*, pages 189–206. Amer. Math. Soc., 1984.

[14] A. Lakhina, J. W. Byers, M. Crovella, and I. Matta. On the geographic location of Internet resources. *IEEE Journal on Selected Areas in Communications, Special Issue on Internet and WWW Measurement, Mapping, and Modeling*, 2003.

[15] H. Lim, J. C. Hou, and C.-H. Choi. Constructing Internet coordinate system based on delay measurement. In *Proceedings of the ACM/SIGCOMM Internet Measurement Conference (IMC-03)*, 2003.

[16] N. Linial. Finite metric spaces — combinatorics, geometry and algorithms. In *Proceedings of the International Congress of Mathematicians III*, pages 573–586, 2002.

[17] N. Linial, E. London, and Y. Rabinovich. The geometry of graphs and some of its algorithmic implications. *Combinatorica*, 15:215–245, 1995.

[18] E. Ng and H. Zhang. Predicting Internet network distance with coordinates-based approaches. In *Proceedings of Infocom*, 2002.

[19] V. N. Padmanabhan and L. Subramanian. An investigation of geographic mapping techniques for Internet hosts. In *In Proceedings of ACM/SIGCOMM '01*, August 2001.

[20] M. Pias, J. Crowcroft, S. Wilbur, S. Bhatti, and T. Harris. Lighthouses for scalable distributed location. In *Second International Workshop on Peer-to-Peer Systems (IPTPS '03)*, Feb 2003.

[21] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, 1988.

[22] RouteViews. http://www.routeviews.org.

[23] The skitter project. http://www.caida.org/tools/measurement/skitter/.

[24] Sockeye Networks. http://www.sockeye.com/.

[25] G. Strang. *Linear Algebra and Its Applications*. Harcourt, Inc., 1988.

[26] H. Tangmunarunkit, R. Govindan, S. Shenker, and D. Estrin. The impact of routing policy on Internet paths. In *Proceedings of INFOCOM 2001*, pages 736–742, 2001.

[27] T. Tankel and Y. Shavitt. Big-bang simulation for embedding network distances in Euclidean space. In *Proceedings of IEEE INFOCOM 2003*, April 2003.

[28] W. S. Torgerson. Multidimensional scaling: I. theory and method. *Psychometrika*, 17:401–419, 1952.

[29] J. Vleugels and R. C. Veltkamp. Efficient image retrieval through vantage objects. *Pattern Recognition*, 35(1):69–80, Jan 2002.

[30] Q. Wong and Z. Wu. A linear-time algorithm for solving the molecular distance geometry problem with exact inter-atomic distances. *Journal of Global Optimization*, pages 365–375, 2002.

[31] F. W. Young and R. M. Hamer. *Multidimensional Scaling: History, Theory, and Applications*. Lawrence Erlbaum Associates, Hilldale, N.J., 1987.