Lighthouses for Scalable Distributed Location

Marcelo Pias[1], Jon Crowcroft[2], Steve Wilbur[1], Tim Harris[2], and Saleem Bhatti[1]

[1] University College London (UCL)
Computer Science Dept.
Gower Street, WC1E 6BT, London, UK
{m.pias,s.wilbur,s.bhatti}@cs.ucl.ac.uk

[2] University of Cambridge
Computer Laboratory
15 JJ Thomson Avenue, CB3 0FD, Cambridge, UK
{jon.crowcroft,tim.harris}@cl.cam.ac.uk

Abstract. This paper introduces *Lighthouse*, a scalable location mechanism for wide-area networks. Unlike existing vector-based systems such as GNP, we show how network-location can be established without using a fixed set of reference points. This lets us avoid the communication bottlenecks and single-points-of-failure that otherwise limit the practicality of such systems.

1 Introduction

Recent years have seen prolific research into large-scale distributed Internet applications drawing on the foundations laid by file-sharing systems, such as Napster, and other *unstructured* peer-to-peer systems such as Gnutella and Freenet (see [11]). This research has developed self-organising content addressable storage based on distributed hash tables (DHT) [15, 12, 7, 2] and distributed trees (DT) [1, 10].

However, the efficiency metrics considered in original versions of these protocols have simply been the number of overlay hops taken while routing a message [13]. This might be appropriate for some very constrained scenarios, but is rarely suitable for realistic deployments. This inflexibility leads to perverse routing policies; a message might be routed in the overlay network via the Europe-US transatlantic link when the two nodes willing to communicate are nearby with a fast local connection, for example, one in London and other in Cambridge.

Distributed network games are another large-scale example which has recently engaged the research community. Among the multiuser games, First person shooters (FPS) are one of the most popular types [17]. In FPS games, network proximity information about the players and servers is an important system requirement. With suitable information, the game discovery mechanism can return a list of servers that are close to a prospective player (e.g. a 'k-Nearest Neighbours' query).

Motivated by the current lack of network proximity information in these systems, we started from the following questions: could we characterise network proximity in a scalable model? Could this model help the systems in selecting appropriate close peers?

Network proximity, in the context of this paper, refers to how close node A is to node B in respect to the underlying IP topology. We characterise it with measures of IP network performance. The propagation delay, for instance, can indicate whether or not two nodes are close neighbours.

To capture the proximity between nodes, we can compute their location in the Internet using a set of coordinates. How we calculate such locations is the main idea of this paper.

We shall now make two definitions before we introduce the the problem of interest. First, a **general space** $\mathbf{M}$ is defined by the pair $(\mathbf{X}, \mathbf{d})$ where $\mathbf{X}$ represents the set of valid objects and $\mathbf{d}$ is a function, either metric or non-metric, that represents the distance between these objects such that $\mathbf{d} : \mathbf{X} \times \mathbf{X} \rightarrow \mathbb{R}$. In contrast, a **vector space** is a set $\mathbf{V}$ that is closed under appropriate vector addition and scalar multiplication operations.

These definitions have a broad scope. A general space represents objects and their mutual distances; whereas a vector space represents objects, their distances and locations. In the Internet case, the space $\mathbf{M}$ may be a set of network nodes (objects) spaced according to a particular network performance metric (distance). For instance, properties such as propagation delay and bandwidth can define two types of distance measures; consequently under certain assumptions, they create two metric spaces.

The problem then is defined as follows. We refer to it as the *mapping* problem and it consists of:

- finding a scalable mapping method to transform *objects* $\{x_1, ..., x_n\}$, in our case, network nodes, of the original space $\mathbf{M}$ onto *points* $\{v_1, ..., v_n\}$ in a target vector space $\mathbf{V^k}$ (k is the dimensionality) in such a way that the distance measures (i.e. delay) are preserved, i.e. $d(\mathbf{x_i}, \mathbf{x_j}) \sim D(\mathbf{v_i}, \mathbf{v_j})$ for $i, j > 0$; where D is another distance function.
- *constraint*: we only know a few distance measures between these objects. This is because we want the system to scale and having a full distance matrix $|X| \times |X|$ is impractical.

The constraint above leads us to use *pivoting* techniques to map the location of an object in a general space onto a vector-space location. These techniques consider the distance from a given object to a number of pre-selected *pivots* $\{p_1, ..., p_n\} \in X$. Pivoting is the common framework for a large class of nearest neighbour algorithms [4, 16].

In this paper we study two distinct techniques that employ pivoting to solve the 'mapping' problem. First, we introduce *absolute* or *global* coordinates-based approaches that always use the same set of well-known pivots. Because of this, such techniques create potential bottlenecks. This is not to mention the consequences when a pivot node becomes unavailable (e.g. pivot failure). The GNP framework [18] is included in this category.

To overcome the issue of well known pivots, we then introduce *Lighthouse*, an alternative technique that uses *relative* or *multiple local* coordinates-systems.

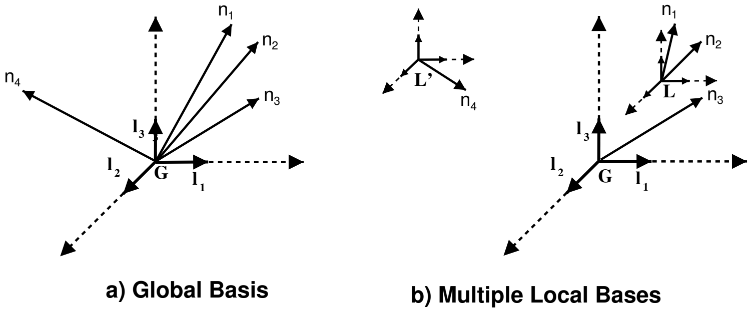

Fig. 1. Global x Multiple Local Bases

Lighthouse is (a) scalable: by relying on any arbitrary set of pivots, it avoids a single set of reference points (pivots) forming bottlenecks; (b) accurate: by solving the *mapping* problem, it devises accurate coordinates for a node.

In what follows, we give an overview of the GNP framework in Section 2, describe the Lighthouse design in more detail in Section 3. We then discuss our initial results and raise a list of questions in Section 4.

2 GNP

The GNP (Global Network Positioning) framework [18] for predicting Internet network distances is based on *absolute* coordinates computed by modelling the Internet as a real vector space. In outline, the GNP architecture is formed from two parts. First, a small set of well known hosts (pivots) called landmarks locate themselves into a real vector space by measuring their mutual distances (delay). These coordinates are taken by hosts that wish to join the system as a global, and therefore unique, basis of the vector space. The landmarks' coordinates are calculated through the solution of a relative error minimization problem: $\sum_{i,j} Error(d_{ij}, \hat{d}_{ij})$ where d_{ij} and $\hat{d}_{ij}$ are the measured and estimated distances between the landmarks i, j.

The second part of the architecture relates to how an arbitrary host calculates its own absolute coordinates based on the landmarks' own coordinates. The joining host measures its round-trip delay to the landmarks and then casts the computation as an overall error minimization problem: $\sum_{i,j} Error(d_{ij}, \hat{d}_{ij})$ where now d_{ij} and $\hat{d}_{ij}$ are the measured and estimated distances respectively from host i to the landmark j; $Error()$ is an error measurement function.

3 Lighthouse

We start by introducing Figure 1a. The basis **G** of this 3-D vector space comprises vectors $\{\mathbf{l_1}, \mathbf{l_2}, \mathbf{l_3}\}$. The second observation is that **G** must be formed by well-known pivot nodes, i.e., the same nodes must be contacted by every joining node. In fact, this is a characteristic of GNP, 'binning' [14] and 'beaconing'

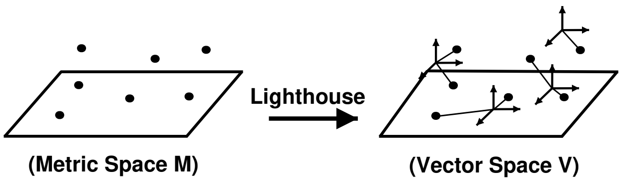

Fig. 2. Lighthouse Overview

[9] frameworks in terms of how they manage reference points. It turns out that this characteristic has the disadvantage that it makes the system not fully self-organised. What happens if the pivot nodes (e.g. landmarks/beacons) are not available at a given instant of time? Who should a joining node contact instead to locate itself in the system?

To overcome the above issue of *well-known pivots*, we present Lighthouse[3], a technique that explores two concepts: multiple local bases together with a transition matrix in vector spaces. Lighthouse allows the flexibility for any host to determine its coordinates relative to any set of pivot nodes provided it maintains a transition matrix. Such a matrix does what the maritime chart does for navigation. It gives a basic instrument for gauging a global position when this is deemed necessary. With the idea of local positioning, better scalability of the system can be achieved. Figure 1b shows a follow-up configuration of the global basis scenario achieved with the Lighthouse framework. Now nodes n_1, n_2, n_3, n_4 are located in different local basis, $\mathbf{L}$ and $\mathbf{L}'$, in a decentralised manner.

Figure 2 presents an example of our technique applied to a 3-D real vector space. Points at the left side plot network nodes as they might be observed in the IP network (metric space $\mathbf{M}$). The right side shows the same points mapped onto a vector space $\mathbf{V}$. With pivoting, we choose arbitrary $\mathbf{k+1}$ local reference points, which we call *lighthouses*. Unlike GNP, our framework relies on a set of nodes from which different joining hosts may select differently. However, each of these hosts has to preserve the invariant: a transition matrix $\mathbf{P}$, which is only applicable to calculating a global position, has to be correctly maintained.

We shall now introduce details of the four step procedure followed by a joining host.

3.1 Finding Lighthouses

The bootstrap of the system occurs as follows:

- **Joining node:** a new node $\mathbf{n_i}$ finds an entry point node $\mathbf{n_j}$, i.e. any node that is already in the system. Node $\mathbf{n_j}$ provides to $\mathbf{n_i}$ a list of nodes that

[3] Historically, lighthouses played a vital role to navigation. The first and most notorious lighthouse, Pharos of Alexandria (Egypt), was built about 270 B.C. When looking at this unique tower with a bright light at the top, a ship's crew can compute their local position relative to it (local reference). Eventually, the position can be transformed into a global one by using maritime charts and the like. Nowadays, GPS (Global Positioning System) with its replicated service has made this method redundant.

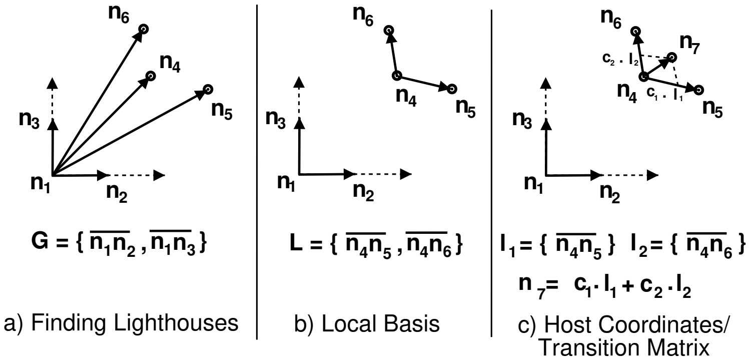

Fig. 3. 2-D Example

can potentially act as $\mathbf{n_i}$ lighthouses. The joining node selects $\mathbf{k+1}$ nodes among those in this list. It then constructs a local basis $\mathbf{L} = \{\mathbf{l_1}, \mathbf{l_2}, ..., \mathbf{l_k}\}$, where each vector $\mathbf{l}$ is a pair of lighthouses. This basis spans the $\mathbf{V^k}$.

- **First nodes:** when $\mathbf{n_i}$ is the m-th node such that $m \leq k+1$, $\mathbf{n_i}$ is considered as one of the first nodes. As $\mathbf{n_i}$ cannot have other $k+1$ lighthouses, it constructs a local basis with the lighthouses that already joined. The idea is to build the first basis after $k+1$ nodes have joined in the system.

Once the joining node $\mathbf{n_i}$ has been given a list of nodes that can act as its lighthouses, it then measures a set of network performance metrics between itself and the lighthouses. The technique by which these measurements are undertaken will vary according to the context. The IDMAPS project [5] found that the propagation delay can be triangulated, so the delay between points (a,c) can be estimated based on the delay between (a,b) and (b,c). As a result, the round-trip time (RTT) measured through ICMP ECHO packets may be a practical tool to incorporate delay as a metric. Additionally, techniques that measure the available bandwidth look promising. However, we have only explored the network delay metric in this paper.

With a $\mathbf{k} \times \mathbf{k}$ matrix of network performance metric values, the joining node computes the coordinates of a local basis $\mathbf{L}$.

Figure 3 introduces a 2-D example. We assume that there are six nodes already in the system: $\{n_1, n_2, n_3, n_4, n_5, n_6\}$ (Figure 3a). Suppose a new node, n_7, wants to join in. As the first step, it contacts a node in the system, say n_4, in order to get a list of lighthouses. In this example, n_4 sends a list of three nodes to act as n_7 lighthouses: $\{n_4, n_5, n_6\}$. At this time, n_7 start measuring the distance, propagation delay, between itself and the three lighthouses.

The following sections describe the method that calculates a local basis $\mathbf{L}$ using the lighthouses.

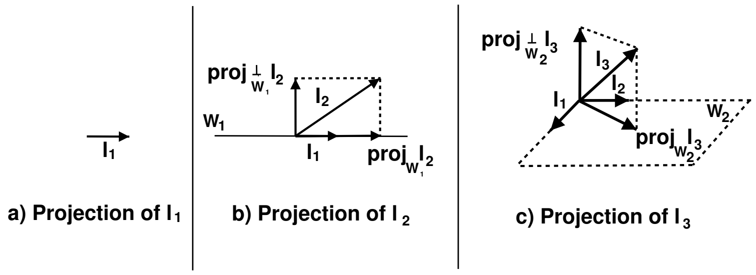

Fig. 4. Gram-Schmidt Process

3.2 Local Basis Coordinates

Any node that wants to take part in the system has to compute its own coordinates relative to a local basis. However, it must first determine the coordinates of the basis that it will be using. To do this, node $\mathbf{n_i}$ calculates $\mathbf{L} = \{\mathbf{l_1}, ..., \mathbf{l_k}\}$ where $\mathbf{l_i}$ is a pair of lighthouse nodes $\overline{n_r n_s}$. It applies the Gram-Schmidt process [3] described as follows.

$$\begin{cases} \mathbf{l_1} &= proj_{w_0}\mathbf{l_1} \ + \ proj^{\perp}_{w_0}\mathbf{l_1}; \\ \mathbf{l_2} &= proj_{w_1}\mathbf{l_2} \ + \ proj^{\perp}_{w_1}\mathbf{l_2}; \\ \quad \vdots & \\ \mathbf{l_k} &= proj_{w_{k-1}}\mathbf{l_k} \ + \ proj^{\perp}_{w_{k-1}}\mathbf{l_{k-1}}. \end{cases} \tag{1}$$

Where $proj_{w_{i-1}}\mathbf{l_i}$ is the projection of $\mathbf{l_i}$ along the finite-dimensional subspace $\mathbf{W_{i-1}}$ of $\mathbf{V^k}$; whereas the vector $proj^{\perp}_{w_0}\mathbf{l_1}$ is called the the component of $\mathbf{l_i}$ orthogonal to $\mathbf{W_{i-1}}$.

We shall explain the Gram-Schmidt process, with a 3-D basis construction example. In the first step (Figure 4a), $\mathbf{l_1}$ is projected into subspace $\mathbf{W_0}$. Vector $\mathbf{l_1}$ now spans the one dimensional subspace $\mathbf{W_1}$. In the second step (Figure 4b), vector $\mathbf{l_2}$ is projected along and orthogonal to $\mathbf{W_1}$. Over the last step (Figure 4c), vector $\mathbf{l_3}$ is calculated as the sum of its component along the subspace $\mathbf{W_2}$, spanned by $\mathbf{l_1}$ and $\mathbf{l_2}$, and by its component orthogonal to $\mathbf{W_2}$.

The joining node, n_7, uses the Gram-Schmidt process to compute a local basis $\mathbf{L} = \{\overline{n_4 n_5}, \overline{n_4 n_6}\}$ (Figure 3b).

3.3 Host Coordinates

At this stage, node $\mathbf{n_i}$ has fresh coordinates of its local basis. It may now calculate its own set of coordinates. However, as a side-effect of choosing arbitrary $\mathbf{k+1}$ lighthouse nodes to span the vector space $\mathbf{V^k}$, it is probable that these vectors will form an alternate basis, not necessarily an orthogonal one. Chances are that the computed basis will be oblique. In that case, we have to be able to use any type of basis (oblique/orthogonal), so that node $\mathbf{n_i}$ coordinate vectors will be a linear combination of the local basis $\mathbf{L}$:

$$\mathbf{n_i} \ = \ c_1\mathbf{l_1} \ + \ c_2\mathbf{l_2} \ + \ \cdots \ + \ c_k\mathbf{l_k} \tag{2}$$

By taking the inner product $<,>$ of both sides of Eq.2 with every vector in $\mathbf{L} = \{\mathbf{l_1}, \mathbf{l_2}, ..., \mathbf{l_k}\}$, we are left with the following system of equations:

$$\begin{cases} <\mathbf{n_i}, \mathbf{l_1}> = <c_1\mathbf{l_1}, \mathbf{l_1}> + \cdots + <c_k\mathbf{l_k}, \mathbf{l_1}> \\ <\mathbf{n_i}, \mathbf{l_2}> = <c_1\mathbf{l_1}, \mathbf{l_2}> + \cdots + <c_k\mathbf{l_k}, \mathbf{l_2}> \\ \qquad\vdots \\ <\mathbf{n_i}, \mathbf{l_k}> = <c_1\mathbf{l_1}, \mathbf{l_k}> + \cdots + <c_k\mathbf{l_k}, \mathbf{l_k}> \end{cases} \tag{3}$$

Solving the system 3, we obtain the scalars $\mathbf{c_i}$. As the only given input to our technique is the distance measures (e.g. delay) from the joining node to a set of lighthouses, an expansion of the system above is essential. Thus, two formulae of the Algebra are required.

$$<\mathbf{u}, \mathbf{v}> = \ \|\mathbf{u}\| \ \|\mathbf{v}\| \ \cos(\widehat{\mathbf{u}, \mathbf{v}}) \tag{4}$$

$$<\mathbf{u}, \mathbf{u}> = \ \|\mathbf{u}\|^2 \tag{5}$$

Formula 4 gives the cosine of the angle between two vectors $\mathbf{u}$ and $\mathbf{v}$; whereas formula 5 is a derivation of the first since the angle θ between identical vectors is 0. Substituting these two formulae in the system 3 yields:

$$\begin{cases} c_1\|\mathbf{l_1}\| + \cdots + c_k\|\mathbf{l_k}\| \cos(\widehat{\mathbf{l_1}, \mathbf{l_k}}) = \|\mathbf{n_i}\| \cos(\widehat{\mathbf{n_i}, \mathbf{l_1}}) \\ \qquad\qquad\qquad\vdots \\ c_1\|\mathbf{l_1}\| \cos(\widehat{\mathbf{l_1}, \mathbf{l_k}}) + \cdots + c_k\|\mathbf{l_k}\| = \|\mathbf{n_i}\| \cos(\widehat{\mathbf{n_i}, \mathbf{l_k}}) \end{cases} \tag{6}$$

In synthesis, the node's coordinates are calculated by solving the system of linear equations (6). Geometrically, this represents the projections of the node distance measures along the vectors of the local basis L.

In our 2-D example (Figure 3c), node n_7 solves a simple linear system in two variables: c_1 and c_2. As a result, the coordinates of n_7 become $c_1.\mathbf{l_1} + c_2.\mathbf{l_2}$, where $\mathbf{l_1} = \overline{n_4 n_5}$ and $\mathbf{l_2} = \overline{n_4 n_6}$.

3.4 Transition Matrix

We allow nodes to arbitrarily choose their lighthouse nodes (local basis) provided they preserve the invariant of rightly maintaining a transition matrix $\mathbf{P}$. The question is how a joining node knows about the global basis $\mathbf{G}$ without measuring any property between itself and the nodes that form such a basis. To answer this question, we bring the idea of *basis changing* into our technique.

If we change the basis for a vector space $\mathbf{V^k}$ from some old basis $\mathbf{B} = \{\mathbf{u_1}, ..., \mathbf{u_k}\}$ to some new basis $\mathbf{B'} = \{\mathbf{u'_1}, ..., \mathbf{u'_k}\}$, then the old coordinate matrix $[\mathbf{v}]_B$ of a vector $\mathbf{v}$ is related to the new coordinate matrix $[\mathbf{v}]_{B'}$ of the same vector by the equation:

$$[\mathbf{v}]_{B'} \ = \ P^{-1} \ [\mathbf{v}]_B \tag{7}$$

Table 1. Key Parameters

Dimensions	Distance	Probes	Tolerance
3	L_2 (Euclidean)	4	10^{-6} (GNP only)

where the columns of **P** are the coordinate matrices of the new basis vectors relative to the old basis, that is, the column vectors of **P** are:

$$P = \begin{bmatrix} [\mathbf{u'_1}]_B \\ \cdots \\ [\mathbf{u'_k}]_B \end{bmatrix} \tag{8}$$

As a result, node $\mathbf{n_i}$ computes a transition matrix **P** between its local basis **L** and the global basis **G**. This does not require any additional distance measurements. The only requirement is that the entry point node $\mathbf{n_j}$ supplies either the coordinates of **G** or its own **P** transition matrix.

The transition matrix P calculated by n_7 (Figure 3c) contains the coordinates of the local basis $\mathbf{L} = \{\mathbf{l_1}, \mathbf{l_2}\}$ relative to the global basis G. This in fact the coordinates of the lighthouses that compose L, i.e., $\{n_4, n_5, n_6\}$. Therefore, n_7 devises **P** with nothing more than the information it already has.

We expect nodes to re-calculate their coordinates from time to time due to frequent network topology changes (i.e. an optical link was shut down). Such changes are captured by the network performance metrics used such as the propagation delay. In this case, a participating node re-computes its coordinates following the four steps above. If for some reason, a lighthouse node becomes unavailable during this re-calculation process, the participating node then chooses an alternate lighthouse to devise the transition matrix.

4 Experimental Evaluation

In this section, we present an initial analysis of our technique. We compared the accuracy of Lighthouse delay estimates against the GNP estimates. Accuracy, in this context, is how close the distance predicted by our technique is to the real distance measured. If we achieve high level of accuracy that means we can compute accurate node locations.

The data used in this experiment was the *global* data set collected by the GNP project[4]. It consists of two matrices with delay measures. The *probe matrix* holds the mutual distance measures between 19 probes. The second matrix, called *target matrix*, contains the delay measures between 869 target hosts and the 19 probes. The delay was measured by ICMP ECHO packets.

Table 1 shows the key parameters used in both implementation of these two techniques. The tolerance parameter was the convergence error of the minimization method used by the GNP code.

[4] Measurement data available at
`http://www-2.cs.cmu.edu/~eugeneng/research/gnp`

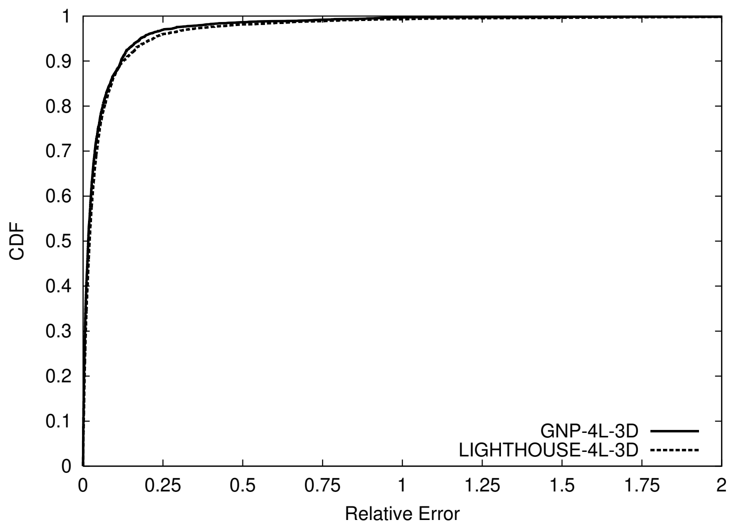

Fig. 5. Accuracy of Lighthouse: Calibration

The strength of Lighthouse, as explored in the previous sections, is its capability of working with multiple local bases through oblique projections. To fairly compare our technique to GNP, we limited the experiment to a single and global basis.

We chose four arbitrary probes among the nineteen to serve as the lighthouses and landmarks nodes. With distance between four probes, Lighthouse code computed a local basis for a 3-D vector space; whereas the GNP code calculated a global solution for the distance error minimization problem.

Moreover, a common framework was required to compare both techniques. Hence, we divided the evaluation in two sub-processes. The first one, called *calibration*, relates to how accurate a technique is when computing the local basis (Lighthouse) or the global basis (GNP). Distances measures between the four chosen probes were required for this sub-process. The *extrapolation*, the second sub-process, tells how accurate a technique is to predicting/extrapolating distance measures between arbitrary nodes. In order to help the comparison two accuracy metrics were used.

$$\text{RelativeError} = \frac{|\text{Measured} - \text{Estimated}|}{\text{Measured}} \tag{9}$$

$$\text{Ratio} = \frac{\text{Estimated}}{\text{Measured}} \tag{10}$$

Formula 10 gives the ratio of an estimated to a measured distance. Ideally, a curve resulting from this metric is a vertical line at x=1. On the other hand, the relative error metric (formula 9) results in zero when the estimated matches the measured distance by 100%.

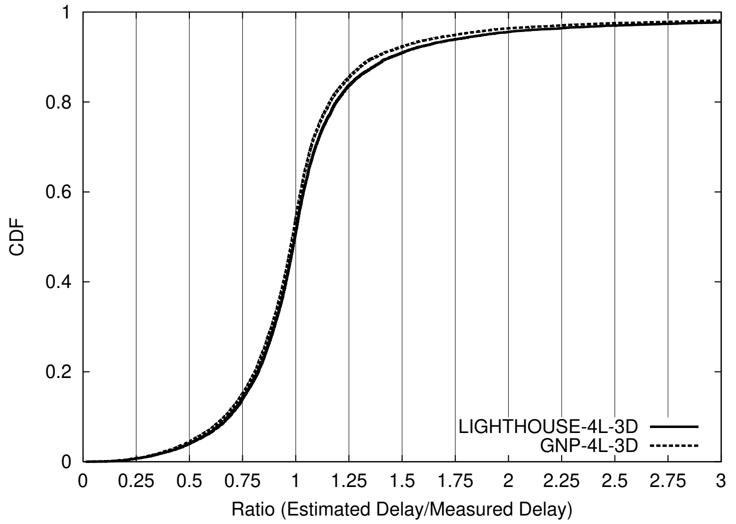

Fig. 6. Accuracy of Lighthouse: Extrapolation

In Figure 5, we plot the CDFs of the relative error of Lighthouse and GNP for the calibration sub-process.

As we expected, both techniques achieved high levels of accuracy measured by their relative errors. We should point out that the measured distances between the four probes should match the distances computed by each technique. This property determines how well the technique can extrapolate distance measures. Lighthouse presented almost the same average accuracy of GNP. Both techniques could estimate 99% of the distances within a relative error of 0.5 or less.

Figure 6 compares the CDFs of the ratios of Lighthouse and GNP delay estimates to those measured. Despite the fact that the two techniques presented equivalent results, Lighthouse was slightly better than GNP for ratios less than 1. On the other hand, 70.34% of GNP estimates were within a 25% error margin as opposed to 69.61% of Lighthouse estimates. As much as 41% of Lighthouse and GNP estimates were within an error of 10%.

Finally, we offer some back-of-the-envelope numerical support for why Lighthouse should scale better than GNP. We used in our experiments 869+19 hosts. Lighthouse could have used any local basis from a combination of 888 hosts taken 4 at a time, i.e. $C(888,4)$. This yields a selection of 25733706090 bases that a joining node can choose as opposed to only one global basis offered by GNP.

4.1 Discussion

GNP represents the first step to modelling Internet distances with a single coordinate system. Lighthouse furthers this by proposing a system that can use multiple coordinate systems. Despite their dissimilarities for solving the same problem, it seems that both methods face similar questions. Some of them are connected to ongoing mathematical refinements in both models. Others are associated with the problem of choosing the right network performance metrics. In this section we raise some questions. Our intention is not to cover the full spectrum of these issues but to ask researchers to look at different perspectives at this problem space.

Network Performance Metrics. So far we have only experimented with Internet delay as it can be triangulated [5]. What are the additional metrics that could be used? How could these metrics be *practically* measured? Is 'available bandwidth' a feasible metric? If so, could we characterise it as we did with network delay?

Distance Function. In our experiments we tested the L_p family of functions with $L_p = \left(\sum_{i=1}^{k} |x_i - y_i|^p\right)^{1/p}$. When $p = 2$ we have the L_2, which is the Euclidean distance. In contrast, for $p = 1$, we have the 'Manhattan' or block distance. Additionally, $p < 1$ results in a non-metric distance function used where the distances do not obey the triangle inequality [8]. We varied p from 0.0 to 6.0 in our experiments and found that the L_2 function has given better delay estimates than any other derivation of L_p. However, the question is: could the L_2 distance function be applied to other network performance metrics such as the available bandwidth?

The Curse of Dimensionality. Network performance metrics may suffer from large differences between their *representational* dimensions k in a vector space and their *intrinsic* dimensionality. This is related to the real number of dimensions that have to be used while maintaining the original distance and it is called *the curse of dimensionality.* To match the *intrinsic* dimensionality of Internet delay distances about 5 to 7 dimensions were required [18] . This prompts the question: do we need to experimentally find out the intrinsic dimensionality of other network performance metrics such as the available bandwidth? Or, could we assume that a vector space with 5 to 7 dimensions can model any network performance metric?

Complex versus Real Scalars. We are investigating the benefits of using complex numbers rather than real scalars. Therefore, a $\mathbb{C}^3$ complex vector space may suit the intrinsic dimensionality of Internet delay. But why should we use complex vector spaces? As Lighthouse relies on projections, usually oblique, it uses the cosine of angles between the projected segments. With a non-metric distance function, we may also be able to model distances that do not obey the triangle inequality. In doing so we turn the metric space into a non-metric space.

On-line versus Off-line Measurements. On-demand measures of network distances might be too costly to be carried out. This has motivated our work. In contrast, King [6], a latency measurement tool based on the DNS infrastructure, measures on-line latency. What are the trade-offs of using an off-line or on-line measurement tool? A comparison of King to Lighthouse may give some clues of which type of technique we can apply in different contexts.

Choosing Lighthouse nodes. Lighthouse nodes form a local basis **L** which spans a vector space **V**. To be sure that **L** is a proper basis, we need to show that the vectors of **L** are linear independent, i.e. every vector in **V** is expressible as a linear combination of the vectors in **L**. There might be cases where the chosen Lighthouse nodes do not form a linear independent basis, therefore, yielding multiple solutions to the system (6). For example, suppose we want to span a 2-D vector space but the three chosen points lie on the same line. They are linear dependent vectors that can only span a 1-D space. To address this issue, the joining node can check locally, during the second step of our algorithm, whether or not the selected Lighthouse nodes form a vector space basis. Such a test consists of making sure that the following matrix **L** has a nonzero determinant:

$$det(L) = \begin{vmatrix} [\mathbf{l_1}] \\ \cdots \\ [\mathbf{l_k}] \end{vmatrix} \neq 0 \tag{11}$$

where the columns of **L** are the coordinate matrices of the basis vectors.

In contrast, GNP requires a similar test as it encounters the same problem. Unlike Lighthouse, this checking cannot be done locally by the joining node but it should be implemented while selecting the GNP global landmarks.

5 Conclusions

In this paper we have presented a technique, called Lighthouse, that maps objects, i.e. nodes and their distance measures such as delay, onto points in a k-dimensional vector space. Our framework avoids the scalability problem of systems that employ 'well-known' pivots as their reference points. Hence, it gives enough flexibility to a joining host in choosing its set of lighthouses. We believe that Lighthouse is accurate as shown by our initial results. With the same information, a 4x4 matrix of distance delay measures, we were able to achieve similar levels of accuracy as GNP with a 3-D vector space. As for future work, we will be investigating the issues raised in the previous section.

6 Acknowledgments

We thank Sugih Jamin for discussions on distance metrics and triangulation on the Internet. We thank Adam Greenhalgh, Socrates Varakliotis, Tolga Uzuner, Andrew Twigg, Igor Sobrado, Arnaud Jacquet, Jose Suruagy, Kennedy Cheng, Senthil Ayyasamy and the anonymous reviewers for their valuable feedback.

References

1. B. Silaghi, S. Bhattacharjee, and P. Keleher. Query Routing in the TerraDir Distributed Directory. In *Submitted for Publication*, 2001.

2. B. Zhao, J. Kubiatowicz, and A. Joseph. Tapestry: An Infrastructure for Fault-Tolerant Wide-Area Location and Routing. In *Technical Report UCB CSD-01-1141, University of California at Berkeley*, November 2001.
3. L. Corwin and R. Szczarba. *Calculus in Vector Spaces: 2nd edition*. Marcel Dekker Inc., 1994.
4. A. Farago, T. Linder, and G. Lugosi. Fast nearest-neighbor search in dissimilarity spaces. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 15(9):957–962, 1993.
5. P. Francis, S. Jamin, C. Jin, D. Raz, Y. Shavitt, and L. Zhang. Idmaps: A global internet host distance estimation service. *IEEE/ACM Trans. on Networking*, 9(5):525–540, 2001.
6. K. P. Gummadi, S. Saroiu, and S. Gribble. King: Estimating latency between arbitrary internet end hosts. In *ACM SIGCOMM Internet Measurement Workshop (IMW'02)*, Marseille, France, November 2002.
7. I. Stoica, R. Morris, D. Karger, F. Kaashoek, and H. Balakrishnan. Chord: A Scalable Peer-to-Peer Lookup Service for Internet Applications. In *ACM SIGCOMM'01 Conference*, San Diego, USA, August 2001.
8. D. Jacobs, D. Weinshall, and Y. Gdalyahu. Classification with non-metric distances: Image retrieval and class representation. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22(6):583–600, 2000.
9. C. Kommareddy, N. Shankar, and B. Bhattacharjee. Finding close friends on the internet. In *ICNP'01*, Riverside (CA), USA, November 2001.
10. M. Freedman and R. Vingralek. Efficient Peer-To-Peer Lookup Based on a Distributed Trie. In *1st International Workshop on Peer-to-Peer Systems (IPTPS'02)*, MIT, Cambridge, USA, March 2002.
11. Andy Oram. Peer-to-peer: Harnessing the power of disruptive technologies. O'Reilly, March 2001.
12. P. Druschel and A. Rowstron. Pastry: Scalable Distributed Object Location and Routing for Large-scale Peer to Peer Systems. In *18th IFIP/ACM Middleware 2001*, November 2001.
13. S. Ratnasamy, , S. Shenker, and I. Stoica. Routing Algorithms for DHTs: Some Open Questions. In *1st International Workshop on Peer-to-Peer Systems (IPTPS'02)*, MIT, Cambridge, USA, March 2002.
14. S. Ratnasamy, M. Handley, R. Karp, and S. Shenker. Topologically-aware Overlay Construction and Server Selection. In *IEEE INFOCOM' 02*, New York, USA, June 2002.
15. S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. A Scalable Content-Addressable Network. In *ACM SIGCOMM'01 Conference*, San Diego, USA, August 2001.
16. M. Shapiro. The choice of reference points in best-match file searching. *Communication of the ACM*, 20(5), 1977.
17. T. Henderson. Observations on Game Server Discovery Mechanisms. In *ACM SIG MULTIMEDIA NetGames 2002: First Workshop on Network and System Support for Games*, Braunschweig, Germany, April 2002.
18. T.S. Eugene Ng and Hui Zhang. Predicting Internet Network Distance with Coordinates-Based Approaches. In *IEEE INFOCOM' 02*, New York, USA, June 2002.