# Optimizing Dimensionality and Accelerating Landmark Positioning for Coordinates Based RTT Predictions

Dragan Milić, Torsten Braun
Institute of Computer Science and Applied Mathematics
University of Bern, Neubrückstrasse 10, 3012 Bern, Switzerland
email: {milic|braun}@iam.unibe.ch
phone: +41 31 631 {5309|4994}

***Abstract*— In this paper we analyze the positioning of landmarks in coordinates-based Internet distance prediction approaches with focus on Global Network Positioning (GNP). We show that one of the major drawbacks of GNP is its computational overhead for a large number of landmarks and dimensions. In our work we identify two factors, which have a great impact on the computational overhead. The first one is being able to determine the optimal number of dimensions for embedding a given set of landmarks into a Euclidean space. The second factor is the selection of a good starting point for minimizing the total error of embedding. We propose an algorithm based on the simplex inequality (a generalized form of the triangle inequality) to extract the optimal number of dimensions based on distance measurements between landmarks. We also provide methods to compute a good starting point for the minimization problem and to reduce the number of variables involved in the minimization. We performed experiments with data obtained from the PlanetLab all-sites-pings experiment to verify the correctness and performance gains of our algorithm. The experimental results show that our enhancements to GNP landmark positioning are able to find the optimal number of dimensions for embedding the landmarks. These enhancements also accelerate the function minimization.**

## I. Introduction

### A. Motivation

The round-trip-time (RTT) is, besides the available bandwidth, one of the most important measurable properties of the Internet communication. Floyd and Fall have identified in their work [1] that the effective bandwidth of a TCP connection is limited by the following properties of the network path: available bandwidth, packet drop probability and RTT. RTT information can be used in many ways, i.e. for selecting application server mirror sites, for optimizing overlay and peer-to-peer networks, and for finding optimal routes.
Given the number of hosts in the Internet, the overhead of RTT measurements, and storage for all host-pairs ($O(n^2)$), it is not feasible to perform and store RTT measurements for all hosts in the Internet. On the other hand, it has been shown [2] that it is possible to represent Internet hosts as points in a virtual $d$-dimensional Euclidean space $\mathcal{S}$ such that the Euclidean distance between two points in such a space (denoted by $d_{\mathcal{S}}$) is a good prediction of the RTT for the corresponding host-pair. Knowing that the RTT can be represented as a distance in a metric space also indicates that in order to position one host in such virtual space, we do not need to measure RTTs to all other hosts in the Internet. Theoretically it would be sufficient for one host to measure "distances" (RTTs) only to $d-1$ other hosts with known positions in order to be able to determine its position. The procedure of determining an unknown position based only on distance measurements is also known as multilateration. By measuring RTTs (distances) to only a few nodes in the Internet we trade the number of measurements needed for the precision of RTT estimation — the more measurements to different hosts we perform, the more accurate the RTT prediction of the system will be.

### B. Related Work

In the last years there have been numerous proposals for RTT prediction schemes based on coordinates [3]–[10]. Basically there are two types of coordinates based RTT prediction schemes: landmarks-based and landmarks-less. The landmarks-based RTT prediction schemes determine coordinates of few fixed reference points (so called landmarks) and use them to determine the positions of all other hosts. Some of them are based on positioning landmarks hosts using function minimization [3]–[5]. The others [8], [9] use the principal component analysis (PCA) to reduce the dimensionality of the landmark RTT measurements and to position the hosts. Landmarks-less RTT prediction schemes such as VIVALDI [6] or S-VIVALDI [7] are based on distributed simulation of physical systems to iteratively reduce the overall error of the host embedding in a Euclidean space. Another interesting approach is presented in [10] where the authors are proposing embedding the Internet hosts in hyperbolic-spaces instead of Euclidean spaces. Such embedding delivers a better RTT prediction service, since a weighted graph representing the Internet can be better projected into a hyperbolic space than into an Euclidean space.
The first and the most promising landmarks-based RTT prediction scheme is GNP [3], [4]. In GNP, $m$ hosts are chosen as landmarks (denoted by $\mathcal{L}_1 \ldots \mathcal{L}_m$). All other Hosts (denoted by $\mathcal{H}$) in the Internet measure RTT distances (denoted by

$\hat{d}_{\mathcal{HL}_1}, \ldots, \hat{d}_{\mathcal{HL}_m}$) to the landmarks and use this information to determine their position in a virtual space. The coordinates of a host in $\mathcal{S}$ (denoted by $\mathcal{C}^{\mathcal{S}}_{\mathcal{H}} := (\mathcal{C}^1_{\mathcal{H}}, \ldots, \mathcal{C}^d_{\mathcal{H}})$) are determined using multilateration. The multilateration itself is based on minimizing the error function $f_e$. In the case of GNP, the least-squares-error function:

$$f_e(\mathcal{C}^{\mathcal{S}}_{\mathcal{H}}) := \sum_{i=1}^{m} (d_{\mathcal{S}}(\mathcal{C}^{\mathcal{S}}_{\mathcal{H}}, \mathcal{C}^{\mathcal{S}}_{\mathcal{L}_i}) - \hat{d}_{\mathcal{HL}_i})^2$$

is used. The function minimization is performed using the Downhill Simplex [11] method.

For calculating $d_{\mathcal{S}}(\mathcal{C}_{\mathcal{H}}, \mathcal{C}_{\mathcal{L}_i})$, the positions of the landmarks in $\mathcal{S}$ ($\mathcal{C}_{\mathcal{L}_1} \ldots \mathcal{C}_{\mathcal{L}_m}$) must be known. In the case of GNP, the coordinates of the landmarks are computed by minimizing the following error function:

$$f_e(\mathcal{C}^{\mathcal{S}}_{\mathcal{L}_1}, \ldots, \mathcal{C}^{\mathcal{S}}_{\mathcal{L}_m}) := \sum_{i,j \in \{1 \ldots m\}, j > i} (d_{\mathcal{S}}(\mathcal{C}^{\mathcal{S}}_{\mathcal{L}_i}, \mathcal{C}^{\mathcal{S}}_{\mathcal{L}_j}) - \hat{d}_{\mathcal{L}_i\mathcal{L}_j})^2 \tag{1}$$

Besides this error function, the authors of GNP propose in the same paper an alternative normalized error function:

$$f'_e(\mathcal{C}^{\mathcal{S}}_{\mathcal{L}_1}, \ldots, \mathcal{C}^{\mathcal{S}}_{\mathcal{L}_m}) := \sum_{i,j \in \{1 \ldots m\}, j > i} \left( \frac{d_{\mathcal{S}}(\mathcal{C}_{\mathcal{L}_i}, \mathcal{C}_{\mathcal{L}_j}) - \hat{d}_{\mathcal{L}_i\mathcal{L}_j}}{\hat{d}_{\mathcal{L}_i\mathcal{L}_j}} \right)^2$$

The reason for proposing using $f'_e$ instead of $f_e$ is that it yields better results when the relative error performance metric defined as:

$$\left| \frac{d_{\mathcal{S}}(\mathcal{C}_{\mathcal{L}_i}, \mathcal{C}_{\mathcal{L}_j}) - \hat{d}_{\mathcal{L}_i\mathcal{L}_j}}{\min\left(d_{\mathcal{S}}(\mathcal{C}_{\mathcal{L}_i}, \mathcal{C}_{\mathcal{L}_j}), \hat{d}_{\mathcal{L}_i\mathcal{L}_j}\right)} \right|$$

is used to evaluate the performance of the RTT predictions.

Whatever landmark positioning strategy is used, it is crucial that the landmark coordinates are as accurate as possible, since the coordinates of all other hosts in the Internet are calculated relative to them. An error in positioning one landmark decreases the performance of the RTT prediction for almost every host in the Internet.

The landmark positioning proposed in GNP has numerous issues — the most severe one is the computational overhead of the function minimization. The function minimization used for GNP (Downhill Simplex) is very computation-intensive. The reason for this is the more than linear growth of the problem space with each additional variable, lack of a “good” starting point (a point in the vicinity of the optimal solution) and, in the case of landmark positions, the infinite number of possible solutions for embedding landmarks, which slows down the convergence of the function minimization.

### C. Overview

In this paper we focus on improving landmark positioning of GNP by reducing the number of possible solutions to a finite number, determining the optimal number of dimensions for the space, in which the landmarks can be positioned, and by providing a good starting point for the function minimization. The structure of this paper is as follows: In the next Section we provide an estimate of the computational complexity depending on the number of dimensions used for positioning landmarks using function minimization. In Section III we provide a formal description for the problem of finding the landmark coordinates. In Section IV we discuss how the infinite number of solutions can be reduced to a finite number and provide an explicit method for constructing a simplex for the given vertices. In Section V we describe the possible violations of the properties of a $d$-dimensional metric space (the simplex inequality) and how those can be detected (using Cayley-Menger determinants). In Section VI we describe algorithms that utilize the results from Sections IV and V to calculate the optimal number of dimensions for representing landmark coordinates, and to compute a good starting point for finding them. In Section VII we evaluate the proposed algorithm by comparing its accuracy, performance and number of dimensions with the equivalent landmark positioning scheme proposed in GNP. In Section VIII we summarize the results of this paper.

## II. Complexity of Function Minimization

The first question to ask is whether there is a scaling problem for GNP’s landmark positioning regarding the number of dimensions used. As we have mentioned, GNP proposed positioning landmarks by minimizing a least squares objective function $f_e$ defined in (1). At the first glance, computing $f_e$ does not depend on the number of dimensions $d$. If we consider the definition of the Euclidean distance between two landmarks $d_{\mathcal{S}}$ (2), it is clear that the computational overhead of $d_{\mathcal{S}}$ is linear ($d_{\mathcal{S}} \in O(d)$), which also means that $f_e \in O(d)$.

$$d_{\mathcal{S}}(\mathcal{C}_{\mathcal{L}_i}, \mathcal{C}_{\mathcal{L}_j}) := \sqrt{\sum_{k=1}^{d} (\mathcal{C}^k_{\mathcal{L}_i} - \mathcal{C}^k_{\mathcal{L}_j})^2} \tag{2}$$

To determine the computational complexity of the function minimization we still have to determine the number of evaluations of $f_e$ needed to perform the function minimization. An analytical approach to find this number is impractical due to the fact that the number of iterations needed heavily depends on the starting point used to perform the minimization. To still be able to roughly determine the computational overhead of the function minimization, we have gathered empirical data and performed a statistical analysis of it.

For the experiment, we have randomly chosen 20 hosts from the Planet-Lab testbed as landmarks. For those 20 landmarks we retrieved the full RTT measurement matrix from the PlanetLab all-sites-pings experiment [12]. For this distance matrix we used the Downhill-Simplex function minimization as proposed in GNP to determine the landmark positions with the number of dimensions $d$ varying between 1 and 19, which is the theoretical maximum of dimensions with 20 landmarks. For each number of dimensions we have performed 50 function minimizations with different randomly chosen starting points. For each function minimization we have recorded the number of evaluations of the objective function $f_e$ performed to obtain the landmark coordinates.

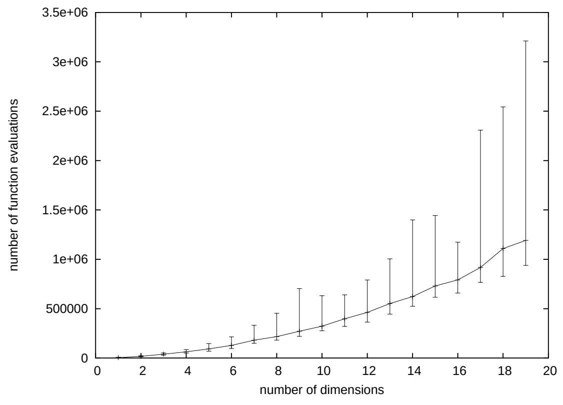

Fig. 1. Number of evaluations of the objective function used to position landmarks

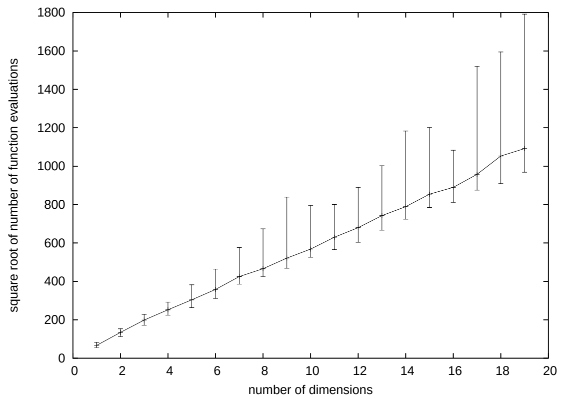

Fig. 2. Square root of the number of evaluations of the objective function used to position landmarks

The results of this experiment are summarized in Fig. 1. This Figure contains the median and the $90\%$ confidence interval of the number of function evaluations performed for each number of dimensions. As we can see, the number of function evaluations performed is increasing more than linearly with the number of dimensions used. Our assumption judging on the shape of the curve is that it increases quadratically. Our assumption is confirmed by Fig. 2 where we have represented the square root of the number of performed function evaluations depending on the number of dimensions. The median values in this Figure increase linearly, meaning that the original function is increasing quadratically.
This result allows us to state that the number of function evaluations is within $O(d^2)$. As shown above, each function evaluation has a linear computational overhead regarding the number of dimensions. Therefore, we can state that the total computational complexity of an average positioning of landmarks depending on the number of dimensions $d$ is cubic ($O(d^3)$).
Fig. 3 shows the average CPU time needed (in seconds) measured in our experiment to find landmark positions depending on the number of dimensions. This Figure clearly demonstrates that reducing the number of dimensions can dramatically

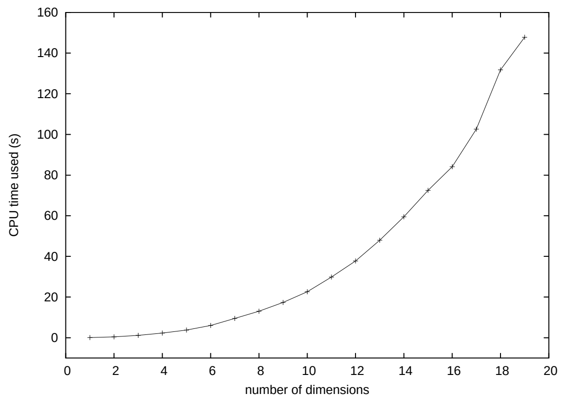

Fig. 3. Average CPU time in a 3 GHz Pentium-D CPU (in seconds) for computing landmark positions depending on the number of dimensions

accelerate the time needed to position the landmarks. For example positioning 20 landmarks in 19 dimensions takes in average 147.795 seconds CPU time. Finding landmark positions for the same data in 6 dimensions takes only 6.045 seconds. In our opinion, results of this experiment justify the need for reducing the number of dimensions used for positioning landmarks.

## III. Landmark Coordinates: the Problem Definition

In GNP we have a set of $m$ landmarks $(\mathcal{L}_1, \ldots, \mathcal{L}_m)$ and a complete set of measured distances between the given landmarks $\{\hat{d}_{\mathcal{L}_i\mathcal{L}_j} : i, j \in \{1, \ldots, m\}\}$, which can be represented as the following distance matrix:

$$\begin{pmatrix} 0 & \hat{d}_{\mathcal{L}_1\mathcal{L}_2} & \cdots & \hat{d}_{\mathcal{L}_1\mathcal{L}_{m-1}} & \hat{d}_{\mathcal{L}_1\mathcal{L}_m} \\ \hat{d}_{\mathcal{L}_2\mathcal{L}_1} & 0 & \cdots & \hat{d}_{\mathcal{L}_2\mathcal{L}_3} & \hat{d}_{\mathcal{L}_2\mathcal{L}_m} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \hat{d}_{\mathcal{L}_{m-1}\mathcal{L}_1} & \hat{d}_{\mathcal{L}_{m-1}\mathcal{L}_2} & \cdots & 0 & \hat{d}_{\mathcal{L}_{m-1}\mathcal{L}_m} \\ \hat{d}_{\mathcal{L}_m\mathcal{L}_1} & \hat{d}_{\mathcal{L}_m\mathcal{L}_2} & \cdots & \hat{d}_{\mathcal{L}_m\mathcal{L}_{m-1}} & 0 \end{pmatrix}$$

We assume that the distance matrix is symmetric with respect to the main diagonal, meaning that $\hat{d}_{\mathcal{L}_i\mathcal{L}_j} = \hat{d}_{\mathcal{L}_j\mathcal{L}_i}$.
For this input, we are interested in finding the landmark coordinates $(\mathcal{C}^{\mathcal{S}}_{\mathcal{L}_1}, \ldots, \mathcal{C}^{\mathcal{S}}_{\mathcal{L}_m})$ in the $d$-dimensional Euclidean space $\mathcal{S} := (\mathbf{R}^d, d_{\mathcal{S}})$. If the distance measurements are error-free, we would be able to construct a $(m-1)$-simplex, the simplest regular figure that can be constructed using $m$ points in an $(m-1)$-dimensional space, with points as the landmarks and side lengths equal to the measured distances between them. It is obvious, that all solutions to our landmark coordinates problem, based on an ideal input, can be obtained by applying isometric operations (operations that preserve distances between the points) such as rotation, translation and reflection to one solution. This means, that in a $(m-1)$-dimensional space there is an infinite number of solutions that represent the positions of a simplex. For example, Fig. 4 shows some of the landmark positioning solutions in two dimensions for the following distances: $(1, 1, \sqrt{2})$. As we can see, any

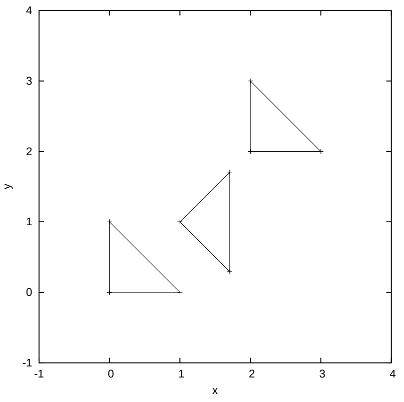

Fig. 4. Example of isometric equivalent solutions for coordinates of a 2-simplex

triangle in Fig. 4 is a solution for the landmark positioning problem. With an increasing number of dimensions0, the problem only gets worse — in addition to rotation, translation and reflection we may also have different shapes of the simplex that have the same distances. Still, the only two isometric operations, which generate an infinite number of solutions, are rotation and translation.
This is actually one of the reasons, why the function minimization used in GNP to determine the landmark coordinates converges very slowly and is non-deterministic.

## IV. Constructing a Simplex

If we have a distance matrix as described in Section III and assuming that it is possible to construct a $(m-1)$-simplex from the input (for a method to verify this assumption see Section V), reconstructing one such simplex is equivalent to finding one solution for the following equation system:

$$\sum_{i=1}^{m-1}(\mathcal{C}^i_{\mathcal{L}_j}-\mathcal{C}^i_{\mathcal{L}_k})^2=(\hat{d}_{\mathcal{L}_j\mathcal{L}_k})^2 \tag{3}$$

where $j \in \{1,\ldots,k\}$ and $k \in \{1,\ldots,m-1\}$

As mentioned in Section III there is an infinite number of possible solutions for landmark positioning, which means that there is an infinite number of solutions for this equation system.

### A. Eliminating Translation and Rotation

To reduce the number of solutions to a finite number, we have to eliminate the isometric operations, which yield an infinite number of solutions. Those operations are rotation and translation.
To eliminate the translation it would be sufficient to require a fixed position of one landmark — for example we could require that the first landmark ($\mathcal{L}_1$) should be positioned in the origin of $\mathcal{S}$: $\mathcal{C}^{\mathcal{S}}_{\mathcal{L}_1}=(0,\ldots,0)$.
Eliminating translation is unfortunately not sufficient. Even if we restrain the first landmark to a fixed position we can obtain an infinite number of solutions by rotating the simplex about an axis which is running trough that fixed point. To eliminate the rotation we can require that the second landmark ($\mathcal{L}_2$) is located on the first axis of the space $\mathcal{S}$, meaning that only the first coordinate of the landmark is variable, the rest of them is zero: $\mathcal{C}^2_{\mathcal{L}_2}=0,\ldots,\mathcal{C}^{m-1}_{\mathcal{L}_2}=0$. The coordinates of the third landmark ($\mathcal{L}_3$) should be located in the plane defined by the first two axes of $\mathcal{S}$ meaning that: $\mathcal{C}^3_{\mathcal{L}_3}=0,\ldots,\mathcal{C}^{m-1}_{\mathcal{L}_3}=0$. For every next landmark we allow one more dimension. The last landmark has no constraints for its coordinates.
By restraining the landmark coordinates we eliminate the possibility of rotation, since by restraining the position of the second landmark to the first axis we eliminate the possibility that the simplex can be rotated around any other axis. Restraining the third landmark to a plane defined by the first two axes eliminates the rotation around the first axis etc. This means, if we have coordinates of landmarks $(\mathcal{L}_1,\ldots,\mathcal{L}_m)$ represented as:

$$\begin{pmatrix} \mathcal{C}^1_{\mathcal{L}_1} & \mathcal{C}^2_{\mathcal{L}_1} & \ldots & \mathcal{C}^{m-2}_{\mathcal{L}_1} & \mathcal{C}^{m-1}_{\mathcal{L}_1} \\ \mathcal{C}^1_{\mathcal{L}_2} & \mathcal{C}^2_{\mathcal{L}_2} & \ldots & \mathcal{C}^{m-2}_{\mathcal{L}_2} & \mathcal{C}^{m-1}_{\mathcal{L}_2} \\ \vdots & \vdots & \ldots & \vdots & \vdots \\ \mathcal{C}^1_{\mathcal{L}_{m-1}} & \mathcal{C}^2_{\mathcal{L}_{m-1}} & \ldots & \mathcal{C}^{m-2}_{\mathcal{L}_{m-1}} & \mathcal{C}^{m-1}_{\mathcal{L}_{m-1}} \\ \mathcal{C}^1_{\mathcal{L}_m} & \mathcal{C}^2_{\mathcal{L}_m} & \ldots & \mathcal{C}^{m-2}_{\mathcal{L}_m} & \mathcal{C}^{m-1}_{\mathcal{L}_m} \end{pmatrix}$$

we reduce the number of unknowns by restraining the coordinates:

$$\begin{pmatrix} 0 & 0 & \ldots & 0 & 0 \\ \mathcal{C}^1_{\mathcal{L}_2} & 0 & \ldots & 0 & 0 \\ \vdots & \vdots & \ldots & \vdots & \vdots \\ \mathcal{C}^1_{\mathcal{L}_{m-1}} & \mathcal{C}^2_{\mathcal{L}_{m-1}} & \ldots & \mathcal{C}^{m-2}_{\mathcal{L}_{m-1}} & 0 \\ \mathcal{C}^1_{\mathcal{L}_m} & \mathcal{C}^2_{\mathcal{L}_m} & \ldots & \mathcal{C}^{m-2}_{\mathcal{L}_m} & \mathcal{C}^{m-1}_{\mathcal{L}_m} \end{pmatrix}$$

Through restraining the landmark coordinates we have eliminated the rotation and translation and also reduced the number of unknowns from $m(m-1)$ to $m(m-1)/2$.
This result alone would noticeably accelerate the convergence of GNP's function minimization; beside that there is an explicit solution to (3), which can be computed directly.

### B. Calculating the Simplex Coordinates

After eliminating rotation and translation, the equation system (3) is simplified to:

$$\begin{aligned} (\mathcal{C}^1_{\mathcal{L}_2})^2 &= (\hat{d}_{\mathcal{L}_1\mathcal{L}_2})^2 \\ (\mathcal{C}^1_{\mathcal{L}_3})^2+(\mathcal{C}^2_{\mathcal{L}_3})^2 &= (\hat{d}_{\mathcal{L}_1\mathcal{L}_3})^2 \\ &\vdots \\ \sum_{i=1}^{m-1}(\mathcal{C}^i_{\mathcal{L}_m})^2 &= (\hat{d}_{\mathcal{L}_1\mathcal{L}_m})^2 \\ (\mathcal{C}^1_{\mathcal{L}_2}-\mathcal{C}^1_{\mathcal{L}_3})^2+(\mathcal{C}^2_{\mathcal{L}_3})^2 &= (\hat{d}_{\mathcal{L}_2\mathcal{L}_3})^2 \\ (\mathcal{C}^1_{\mathcal{L}_2}-\mathcal{C}^1_{\mathcal{L}_4})^2+(\mathcal{C}^2_{\mathcal{L}_4})^2+(\mathcal{C}^3_{\mathcal{L}_4})^2 &= (\hat{d}_{\mathcal{L}_2\mathcal{L}_4})^2 \\ &\vdots \\ \sum_{i=1}^{m-1}(\mathcal{C}^i_{\mathcal{L}_{m-1}}-\mathcal{C}^i_{\mathcal{L}_m})^2 &= (\hat{d}_{\mathcal{L}_{m-1}\mathcal{L}_m})^2 \end{aligned}$$

As mentioned in Section IV-A, the position of the first landmark is set to the origin of $\mathcal{S}$. Since the second landmark is positioned on the first-dimension axis, the following equation delivers the first coordinate of the landmark:

$$\mathcal{C}^1_{\mathcal{L}_2} = \pm\hat{d}_{\mathcal{L}_1\mathcal{L}_2}$$

To determine the position of the third landmark, we have the following equations:

$$(\mathcal{C}^1_{\mathcal{L}_3})^2 + (\mathcal{C}^2_{\mathcal{L}_3})^2 = (\hat{d}_{\mathcal{L}_1\mathcal{L}_3})^2 \quad (4)$$

$$(\mathcal{C}^1_{\mathcal{L}_2} - \mathcal{C}^1_{\mathcal{L}_3})^2 + (\mathcal{C}^2_{\mathcal{L}_3})^2 = (\hat{d}_{\mathcal{L}_2\mathcal{L}_3})^2 \quad (5)$$

by subtracting (4) from (5) we obtain the first coordinate of $\mathcal{L}_3$:

$$\mathcal{C}^1_{\mathcal{L}_3} = -\frac{(\hat{d}_{\mathcal{L}_2\mathcal{L}_3})^2 - (\hat{d}_{\mathcal{L}_1\mathcal{L}_3})^2 - (\mathcal{C}^1_{\mathcal{L}_2})^2}{2\mathcal{C}^1_{\mathcal{L}_2}}$$

The second coordinate of $\mathcal{L}_3$ can be obtained by replacing the computed value of $c^1_{\mathcal{L}_3}$ in (4):

$$\mathcal{C}^2_{\mathcal{L}_3} = \pm\sqrt{(\hat{d}_{\mathcal{L}_1\mathcal{L}_3})^2 - (\mathcal{C}^1_{\mathcal{L}_3})^2}$$

In general, if we have determined the coordinates of the first $k$ landmarks $(\mathcal{C}^{\mathcal{S}}_{\mathcal{L}_1}, \ldots, \mathcal{C}^{\mathcal{S}}_{\mathcal{L}_k})$, we determine the position $\mathcal{C}^{\mathcal{S}}_{\mathcal{L}_{k+1}} := (\mathcal{C}^1_{\mathcal{L}_{k+1}}, \ldots, \mathcal{C}^k_{\mathcal{L}_{k+1}}, 0, \ldots, 0)$ of landmark $\mathcal{L}_{k+1}$ by solving the following equation system:

$$\sum_{i=1}^{k}(\mathcal{C}^i_{\mathcal{L}_{k+1}})^2 = (\hat{d}_{\mathcal{L}_1\mathcal{L}_{k+1}})^2 \quad (6)$$

$$(\mathcal{C}^1_{\mathcal{L}_2} - \mathcal{C}^1_{\mathcal{L}_{k+1}})^2 + \sum_{i=2}^{m-1}(\mathcal{C}^i_{\mathcal{L}_{k+1}})^2 = (\hat{d}_{\mathcal{L}_2\mathcal{L}_{k+1}})^2 \quad (7)$$

$$\begin{aligned}(\mathcal{C}^1_{\mathcal{L}_3} - \mathcal{C}^1_{\mathcal{L}_{k+1}})^2 &+ \\ +(\mathcal{C}^2_{\mathcal{L}_3} - \mathcal{C}^2_{\mathcal{L}_{k+1}})^2 &+ \\ +\textstyle\sum_{i=3}^{k}(\mathcal{C}^i_{\mathcal{L}_{k+1}})^2 &\end{aligned} = (\hat{d}_{\mathcal{L}_2\mathcal{L}_{k+1}})^2 \quad (8)$$

$$\vdots$$

$$\sum_{i=1}^{k-2}(\mathcal{C}^i_{\mathcal{L}_k} - \mathcal{C}^i_{\mathcal{L}_k})^2 + (\mathcal{C}^{k-1}_{\mathcal{L}_{k+1}})^2 = (\hat{d}_{\mathcal{L}_k\mathcal{L}_{k+1}})^2$$

To calculate $\mathcal{C}^1_{\mathcal{L}_k}$ we can subtract (6) from (7). After the subtraction, the result is:

$$\mathcal{C}^1_{\mathcal{L}_{k+1}} = \frac{(\hat{d}_{\mathcal{L}_1\mathcal{L}_{k+1}})^2 - (\hat{d}_{\mathcal{L}_2\mathcal{L}_{k+1}})^2 + (\mathcal{C}^1_{\mathcal{L}_2})^2}{2\mathcal{C}^1_{\mathcal{L}_2}}$$

To calculate $\mathcal{C}^2_{\mathcal{L}_k}$ we can subtract (6) from (8), which leads to:

$$\begin{aligned}\mathcal{C}^2_{\mathcal{L}_{k+1}} = \; & (2\mathcal{C}^2_{\mathcal{L}_3})^{-1}\big[(\hat{d}_{\mathcal{L}_1\mathcal{L}_{k+1}})^2 - (\hat{d}_{\mathcal{L}_3\mathcal{L}_{k+1}})^2 - \\ & -2\mathcal{C}^1_{\mathcal{L}_3}\mathcal{C}^1_{\mathcal{L}_{k+1}} + (\mathcal{C}^1_{\mathcal{L}_3})^2 + (\mathcal{C}^2_{\mathcal{L}_3})^2\big]\end{aligned} \quad (9)$$

This procedure can be used for all coordinates $\mathcal{C}^1_{\mathcal{L}_{k+1}}, \ldots, \mathcal{C}^{k-1}_{\mathcal{L}_{k+1}}$. For the last coordinate $\mathcal{C}^k_{\mathcal{L}_{k+1}}$ we can use the computed values and replace them in (6):

$$\mathcal{C}^k_{\mathcal{L}_{k+1}} = \pm\sqrt{(\hat{d}_{\mathcal{L}_1\mathcal{L}_{k+1}})^2 - \sum_{i=1}^{k-1}(\mathcal{C}^i_{\mathcal{L}_{k+1}})^2}$$

The generalized formula for calculating the simplex coordinates derived from this procedure is:

$$\mathcal{C}^j_{\mathcal{L}_i} = \begin{cases} 0 & , j \geq i \\ \begin{aligned}&(2\mathcal{C}^j_{\mathcal{L}_{j+1}})^{-1}\big[(\hat{d}_{\mathcal{L}_1\mathcal{L}_i})^2 - \\ &-(\hat{d}_{\mathcal{L}_{j+1}\mathcal{L}_i})^2 - \\ &-\sum_{k=1}^{j-1} 2(\mathcal{C}^k_{\mathcal{L}_{j+1}})(\mathcal{C}^k_{\mathcal{L}_i}) + \\ &+\sum_{k=1}^{j}(\mathcal{C}^k_{\mathcal{L}_{j+1}})^2\big]\end{aligned} & , j < i-1 \\ \pm\sqrt{(\hat{d}_{\mathcal{L}_1\mathcal{L}_i})^2 - \sum_{k=1}^{j-1}(\mathcal{C}^k_{\mathcal{L}_i})^2} & , j = i-1 \end{cases} \quad (10)$$

## V. Distances and Metric Spaces

In distances derived from real-world measurements, such as distances to GPS satellites, we are usually confronted with values, which make the exact positioning impossible due to the measurement errors. Still, we know that the measurements stem from a three-dimensional environment, in which the triangle inequality holds. The problem in the coordinates-based RTT prediction is that we are fitting the "distance" measurements into a $d$-dimensional Euclidean space without knowing in advance how many dimensions the space has (the value of $d$).

In GNP, the authors propose the "optimal" number of dimensions and landmarks that can be used based on results of their experiments. In this Section we show that there is a method to determine the optimal number of dimensions for embedding landmarks based only on a distance matrix.

### A. Constraints on Constructing a Simplex

As mentioned earlier, it is not possible to reconstruct a $(m-1)$-dimensional simplex for an arbitrary distance matrix representing distances between $m$ points. The reason for this is that projecting a distance matrix into a Euclidean space (equivalent to constructing a simplex from a distance matrix) poses strict requirements on the distances. To comprehend those requirements, we must recall the definition of a metric space and a Euclidean space.

A metric space is a 2-tuple $(A, d_{\mathcal{S}})$ where $A$ is a set and $d_{\mathcal{S}} : A \times A \to \mathbf{R}$ a distance function. A function $d_{\mathcal{S}}$ is called a distance function if the following conditions are fulfilled for all $a, b, c \in A$:

$$\begin{aligned} d_{\mathcal{S}}(a,b) &\geq 0 \\ d_{\mathcal{S}}(a,b) = 0 &\Leftrightarrow a = b \\ d_{\mathcal{S}}(a,b) &= d_{\mathcal{S}}(b,a) \\ d_{\mathcal{S}}(a,c) &\leq d_{\mathcal{S}}(a,b) + d_{\mathcal{S}}(b,c) \end{aligned} \quad (11)$$

A $d$-dimensional Euclidean space is a special case of a metric space, where:

$$A := \mathbf{R}^d$$

$$d_{\mathcal{S}}(a,b) := \sqrt{\sum_{i=1}^{d}(a_i - b_i)^2}$$

The inequality (11) is also known as the triangle inequality. The semantics of the triangle inequality is that there is no triplet of elements $a, b, c \in A$ where the distance between $a$ and $c$ is greater than the sum of the distance between $a$ and $b$ and the distance between $b$ and $c$. In an Euclidean space, the triangle inequality means that with distances between three points we can always construct a triangle (a 2-simplex) including the pathological case where the triangle is one line segment ($a + b = c$). If we have three distances, which do not fulfill the triangle inequality (for example: a=1, b=3 and c=5), we will not be able to reconstruct a triangle, since there are no three points in any metric space, where the distances would match.

## B. Handling Triangle Inequality Violations

To still be able to have a set of landmarks, one must reduce the number of dimensions. For example, it is obvious that for the following distances: a=1, b=3 and c=5 the triangle inequality does not hold ($5 > 3 + 1$).
To calculate positions of landmarks we can try to reduce the number of dimensions: instead of constructing a 2-simplex (a triangle - which is impossible since the triangle inequality does not hold) we could construct a 1-simplex (a line segment). This 1-simplex can be used as a "base" for positioning the third landmark. The third landmark is positioned in such a way that the embedding-error (e.g. the square error between given and calculated distances) will be minimized. For example, we could assign the following coordinates to the landmarks:$\mathcal{C}^1_{\mathcal{L}_1} = 0$, $\mathcal{C}^1_{\mathcal{L}_2} = 1$ and $\mathcal{C}^1_{\mathcal{L}_3} = 3.5$. This solution is not the only one; actually there are three possible solutions — depending on which pair of landmarks is chosen as the "base". Also worth noting is that the possible solutions are not equal regarding the total square error of such an embedding. Since we are interested in minimizing the total error, the optimal choice would be to use the base, for which the total error is the smallest — in this case the 1-simplex, which is formed by landmarks $\mathcal{L}_1$ and $\mathcal{L}_3$. Whatever solution is chosen, it is usually not an optimal one, since we were not trying to optimize by changing all distance. Still, it is a very good starting point for further minimization — for example using the Downhill Simplex method.

## C. Handling Simplex Inequality Violations

Unfortunately, the triangle inequality is not the only constraint to construct a simplex. For example, for the following distance matrix:

$$\begin{pmatrix} 0 & 3 & 3 & 1.6 \\ 3 & 0 & 3 & 1.6 \\ 3 & 3 & 0 & 1.6 \\ 1.6 & 1.6 & 1.6 & 0 \end{pmatrix}$$

the triangle inequality holds for every triple of distances between the landmarks ($3 \leq 3 + 3, \quad 3 \leq 3 + 1.6, \quad 1.6 \leq 3 + 1.6, \quad 1.6 \leq 3 + 3$). Still we are not able to construct a 3-simplex (tetrahedron) with the given distances, because the triangle inequality is the necessary but not sufficient condition for constructing a $d$-simplex ($d \geq 2$). The sufficient condition in three dimensions would be that the area of every tetrahedron side (triangle) must be smaller than or equal to the sum of the areas of all other sides — this inequality does not hold for our distance matrix.
A more generalized recursive inequality can also be defined for any $d$-simplex: the hyper-volume of every "side" (a $(d-1)$-simplex) of a simplex must be smaller or equal to the sum of hypervolumes of all other simplex-"sides". This inequality is also known as the simplex inequality [13]. The simplex inequality is actually recursive, since, to be able to compute the hyper-volume of a simplex, one must be able to construct a simplex. The recursion ends with the triangle inequality for a 2-simplex. By knowing this, we can construct the landmark coordinates by reducing the dimensionality as described in Section V-B. In this case the "base" is a 2-simplex (a triangle) and the fourth landmark is positioned in the plane defined by the positions of the "base" points.

## D. The Cayley-Menger Determinant

Now that we know the sufficient condition for constructing a simplex, we should be able to find a set of landmarks that can be used as a base. Unfortunately, there is one small detail of the simplex inequality, which is still not trivial. The nontrivial detail is: how do we calculate the hyper-volume of an $(d-1)$-simplex based only on the vertices of the simplex?
In the case of a 1-simplex or a 2-simplex the solution is very simple: a hyper-volume of a line segment is its length, the hyper-volume of a triangle is its area, which can be calculated using Heron's formula. Even for a tetrahedron (a 3-simplex) there is Tartaglia's formula, but still we need a general approach, which delivers the volume of a $d$-simplex, e.g., the Cayley-Menger Determinant [14].
The Cayley-Menger determinant delivers the square of the hyper-volume of a $d$-dimensional simplex $S_d$: $V^2(S_d)$ based only on vertices of the simplex. The square hyper-volume of $S_d$ is defined as

$$V^2(S_d) = \frac{(-1)^{d+1}}{2d(d!)^2}\det(B) \tag{12}$$

where $B$ is a matrix obtained by bordering a quadratic distance matrix with a top row $(0, 1, \ldots, 1)$ and a left column $(0, 1, \ldots, 1)^T$. For example for a 3-simplex the $B$ matrix would look like:

$$B = \begin{pmatrix} 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & (\hat{d}_{\mathcal{L}_1\mathcal{L}_2})^2 & (\hat{d}_{\mathcal{L}_1\mathcal{L}_3})^2 & (\hat{d}_{\mathcal{L}_1\mathcal{L}_4})^2 \\ 1 & (\hat{d}_{\mathcal{L}_2\mathcal{L}_1})^2 & 0 & (\hat{d}_{\mathcal{L}_2\mathcal{L}_3})^2 & (\hat{d}_{\mathcal{L}_2\mathcal{L}_4})^2 \\ 1 & (\hat{d}_{\mathcal{L}_3\mathcal{L}_1})^2 & (\hat{d}_{\mathcal{L}_3\mathcal{L}_2})^2 & 0 & (\hat{d}_{\mathcal{L}_3\mathcal{L}_4})^2 \\ 1 & (\hat{d}_{\mathcal{L}_4\mathcal{L}_1})^2 & (\hat{d}_{\mathcal{L}_4\mathcal{L}_2})^2 & (\hat{d}_{\mathcal{L}_4\mathcal{L}_3})^2 & 0 \end{pmatrix}$$

The reason why the Cayley-Menger determinant is very convenient for our cause lies in the definition of the hyper-volume of a simplex: a volume of a simplex is proportional to the volume of a $d$-dimensional parallelepiped with the same vertices. The hyper-volume of a $d$-dimensional parallelepiped is defined as the parallelepiped's height in one dimension multiplied with the hyper-volume of the parallelepiped's base. The square of a hyper-volume delivers not only the information about the simplex hyper-volume, but also the information about the "heights" of the simplex in lower dimensions.

This information is necessary to be able to draw conclusions about the co-planarity (in the sense of hyper-planes) of a simplex. It can be used to determine the optimal number of dimensions, which should be utilized to construct the simplex. For example: if we calculate the square hyper-volume for the distances of the example from Section V-C we would obtain a negative value.

The negative value of the square hyper-volume means that the volume of the simplex can only be represented as a complex number. It also indicates that at least one "height" of the simplex must be negative, meaning that we will not be able to construct an optimal simplex. The zero value for the square volume suggests that at least one height of the simplex is zero, which means that the points are coplanar (in the sense of hyper-planes) and can be represented with at least one dimension less than $d$. The positive square hyper-volume indicates that we possibly *could* construct an ideal simplex — it does not show if we *can* for sure, because of the possibility that there is an even number of negative heights in the sub-simplexes (the product of an even number of negative numbers is positive). For example, the Cayley-Menger determinant for the following distance matrix computes a positive square hyper-volume, although the triangle equation does not hold $(77 > 50 + 25)$:

$$\begin{pmatrix} 0 & 50 & 77 & 36 \\ 50 & 0 & 25 & 5 \\ 77 & 25 & 0 & 35 \\ 36 & 5 & 35 & 0 \end{pmatrix}$$

This is the reason why we state that the positive square hyper-volume only indicates the possibility to construct a hyper-volume. To be sure one has to recursively check if all hyper-volumes of all sub-simplexes (simplexes in lower dimensions that can be built using a subset of the simplex points) are positive.

## VI. Fast Landmark Positioning

The results presented in Sections IV and V allow us to define two algorithms. Algorithm 1 determines an optimal number for embedding the landmarks by finding all maximal subsets of landmarks, which allow constructing an ideal simplex. Algorithm 2 uses the result of Algorithm 1 to find a good starting point for the GNP's function minimization by using the "best" simplex as a base for embedding. The input of algorithms is a full distance matrix, which contains RTT measurements between all landmarks. The output is the number of dimensions and a good starting point for the function minimization. At the beginning of the Algorithm 1, all possible 1-simplexes (zero dimensional simplexes containing one landmark) are added to the set of current simplexes. At each step of the algorithm, the set of current simplexes is extended by adding new simplexes. A new simplex is added to the set of current simplexes by trying to add one landmark to each simplex from the set and calculating the the Cayley-Menger determinant such a new simplex. If the value of the determinant is positive the new simplex is added to the set of the current simplexes. After all possible simplexes are generated, simplexes with the maximal number of landmarks are chosen and passed to the Algorithm 2.

Algorithm 2 determines a good starting point for the function minimization. Output of the algorithm are coordinates of landmarks. The algorithm processes each simplex obtained from Algorithm 1 the same way: The position of all landmarks that are within one of the simplexes are positioned using (10). All other landmarks are positioned using the GNP's host positioning. By doing so, we obtain a set of possible landmark positions. From this set our algorithm chooses the landmark positions with the lowest embedding error as the return value.

**Algorithm 1**: *FindMaxSimplexes*: algorithm for finding all simplexes of the maximal dimensionality

```
Input  : Distance matrix $D$ containing all distances
         between $m$ landmarks
Output: A set $S$, which contains all simplexes (each of
         them represented as a set of indexes) that can
         be constructed from $D$
$S \leftarrow \{\{x\}|x \in \{1, \cdots, m\}\}$;
for $i \leftarrow 1$ to $m$ do
    for $x \in S$ do
        /* Add a new simplex if its square
           volume is positive                    */
        if SimplexQuadVolume($D, x \cup \{i\}$)$> 0$ then
            $S \leftarrow S \cup \{x \cup \{i\}\}$ ;
        end
    end
    /* Find the maximal simplex size             */
    max $\leftarrow \max(\|x\| : x \in S)$;
    /* Remove all simplexes that cannot
       grow larger than the maximal
       simplex                                   */
    $S \leftarrow S \backslash \{x : x \in S, \|x\| + (m - i) <$ max$\}$;
end
return $S$;
```

**Algorithm 2**: *FindLandmarkPositions*: algorithm for fast landmark positioning

```
Input  : Distance matrix $D$ containing all distance
         between $m$ landmarks
Output: A good starting point for function minimization
         $P$ and the optimal number of dimensions

find all simplexes with the maximal dimensionality using
the FindMaxSimplexes algorithm;
for each found simplex do
    determine the coordinates of the landmarks that are
    in the simplex using (10);
    determine the coordinates of the remaining landmarks
    using GNP host positioning, where the hosts for the
    simplex are used as the landmarks;
    add the positions of the landmarks to the set of the
    potential start-points;
end
choose the best start-point among the potential start
points by finding the one with the smallest total square
error for the embedding;
```

## VII. Evaluation

### A. Evaluation Scenarios

In this Section we present the experimental results of comparing our Fast Landmark Positioning algorithm with the landmark positioning proposed in GNP. In the comparison we are focusing on the accuracy of the landmark positioning (the value of the error function) and the computational overhead (CPU time spent by the process computing the landmark positions). We base our comparison on data obtained from the PlanetLab "all-sites-pings" experiment [12]. To be able to obtain comparable results, we implemented the GNP algorithm and both of our algorithms in the PERL programming language. For the function minimization we used the Math::Amoeba Perl module obtained from CPAN [15], which implements the Simplex Downhill algorithm. For each function minimization we used following parameters: $f_{\text{tol}} = 10^{-7}$ and a maximal eval count of $40000$. All execution time measurements were performed on a computer with a 3 GHz Pentium 4 CPU running the Linux operating system. The methodology for comparison is always the same: we vary the number of landmarks between 4 and 20. For each number of landmarks we randomly choose 20 different subsets of hosts out of total 125 that we have in our data set, which will be used as the landmarks (landmark-set). For each such landmark-set we calculate the coordinates of the landmarks and record the execution time of the process using the UNIX time utility. For each landmark-set we have calculated landmark coordinates using each of the following landmark-positioning:

- *MAX*: The original GNP landmark positioning where $m$ landmarks are positioned in a $(m-1)$-dimensional space. The starting point for the function minimization is randomly chosen. The result of this landmark positioning should have the smallest total positioning error, since we have the maximal number of dimensions available. This positioning is also the slowest one, since we do not have a reasonable starting point for the function minimization and the number of variables is the maximum $(m(m-1))$.
- *OPTDIM*: The GNP landmark positioning where $m$ landmarks are positioned in a $d$-dimensional space. The optimal number of dimensions $d$ is determined using Algorithm 1. The starting point for the function minimization is randomly chosen. This landmark positioning should be much faster than *MAX* since the optimal number of dimensions is usually much lower than the maximal number of dimensions
- *OPTDIM-SP*: The same landmark positioning as *OPTDIM* with a starting point for the function minimization determined by Algorithm 2. The execution time for calculating the starting point is also a part of the total execution time. If our assumption is correct, this algorithm should be faster than *OPTDIM* since the function minimization should be faster if a good starting point is provided.
- *OPTDIM-SP-R*: This landmark positioning algorithm does the same function minimization as *OPTDIM-SP* but eliminates the rotation and translation of the resulting positions (for details see Section IV-A). Our guess was that this algorithm should be the fastest one, since it utilizes every optimization proposed in this paper: optimal number of dimensions, starting point for the function minimization and reducing the number of variables used in the function minimization.

Since there are two flavors of GNP (for details see Section I-B), we perform our test using both alternative error functions for GNP. We refer to GNP using the standard distance square error function $f_e$ as plain GNP. To GNP using the square error of normalized distances we refer as normalized GNP.

### B. Correctness

At the beginning of our evaluation we were interested if our algorithm really finds an optimal number of dimensions. To verify this, we compare for each number of landmarks $m$, the total error of the landmark coordinates computed by *OPTDIM-SP* with the total error that is obtained by performing GNP using $m-1$ dimensions (*MAX*). If our algorithm is correct, the total error of our landmark positioning *OPTDIM-SP* should be equal to or better than the total error of *MAX*. Since the expected value of the total square error varies greatly, depending on the landmark choice, we performed Wilcoxon signed-rank [16] statistical tests for each number of landmarks. The reason for choosing the Wilcoxon signed-rank test is that we have paired measurements (total error of landmark positioning for the same set of landmarks for each algorithm) and that the test makes no assumption about the distribution of the data.

With the test we are trying to detect whether the medians of the total errors of *MAX* and *OPTDIM-SP* are different. To be able to detect all kind of differences we performed three tests for each pair of value sets. The first test ($t_<$) tests whether the median of *MAX* is lower than the median of *OPTDIM-SP*. In this case our conservative hypothesis is $H_0^< : \mu(\mathit{MAX}) \geq \mu(\mathit{OPTDIM\text{-}SP})$. The alternative hypothesis is $H_1^< : \mu(\mathit{MAX}) < \mu(\mathit{OPTDIM\text{-}SP})$. The second test ($t_>$) determines whether the median of *MAX* is higher than the median of *OPTDIM-SP* with the hypotheses: $H_0^> : \mu(\mathit{MAX}) \leq \mu(\mathit{OPTDIM\text{-}SP})$ and $H_1^> : \mu(\mathit{MAX}) > \mu(\mathit{OPTDIM\text{-}SP})$. The third test ($t_{\neq}$) checks whether the medians of *MAX* and *OPTDIM-SP* significantly differ. In this test we have the hypotheses: $H_0^{\neq} : \mu(\mathit{MAX}) = \mu(\mathit{OPTDIM\text{-}SP})$ and $H_1^{\neq} : \mu(\mathit{MAX}) \neq \mu(\mathit{OPTDIM\text{-}SP})$. To determine which hypothesis is correct, we have to compare the $p$ values with the significance level ($\alpha = 0.05$).

For example, if we obtain the following $p$ values as depicted in Table I for 5 landmarks and plain GNP: $0.978$ for $t_>$, $0.022$ for $t_<$ and $0.044$ for $t_{\neq}$, we can say that $\mu(\mathit{MAX})$ is significantly higher than $\mu(\mathit{OPTDIM\text{-}SP})$. The reason for this is that we have $p$ values for the tests $t_>$ and $t_{\neq}$ under the significance threshold ($0.05$), meaning that those tests show that the hypotheses $H_1^>$ and $H_1^{\neq}$ are correct.

We have performed the statistical tests for each number of landmarks and for both flavors of GNP (plain and normalized). We have computed the $p$ values of the tests using the *wilcoxon_test()* statistical function of the octave [17] mathematical software. The $p$ values of tests are represented in Table I. As we can see, our algorithm for finding an optimal number

TABLE I

WILCOXON MATCHED PAIRS SIGNED RANK TEST $p$ VALUES OF THE TOTAL ERROR FOR MAX AND OPTDIM LANDMARK POSITIONING.

| | MAX vs OPTDIM-SP (plain GNP) | | | MAX vs.OPTDIM-SP (normalized GNP) | | |
|---|---|---|---|---|---|---|
| | $p(t_<)$ | $p(t_>)$ | $p(t_{\neq})$ | $p(t_<)$ | $p(t_>)$ | $p(t_{\neq})$ |
| 4 | 0.617 | 0.383 | 0.765 | 0.206 | 0.794 | 0.411 |
| 5 | 0.978 | 0.022 | 0.044 | 0.999 | 0.001 | 0.002 |
| 6 | 0.997 | 0.003 | 0.005 | 0.999 | 0.001 | 0.001 |
| 7 | 1.000 | 0.000 | 0.001 | 0.997 | 0.003 | 0.006 |
| 8 | 1.000 | 0.000 | 0.001 | 1.000 | 0.000 | 0.000 |
| 9 | 1.000 | 0.000 | 0.001 | 0.996 | 0.004 | 0.009 |
| 10 | 1.000 | 0.000 | 0.000 | 1.000 | 0.000 | 0.000 |
| 11 | 0.999 | 0.001 | 0.002 | 1.000 | 0.000 | 0.000 |
| 12 | 1.000 | 0.000 | 0.000 | 0.999 | 0.001 | 0.001 |
| 13 | 0.999 | 0.001 | 0.001 | 0.999 | 0.001 | 0.001 |
| 14 | 1.000 | 0.000 | 0.001 | 1.000 | 0.000 | 0.000 |
| 15 | 1.000 | 0.000 | 0.000 | 1.000 | 0.000 | 0.000 |
| 16 | 1.000 | 0.000 | 0.000 | 1.000 | 0.000 | 0.000 |
| 17 | 1.000 | 0.000 | 0.000 | 1.000 | 0.000 | 0.000 |
| 18 | 1.000 | 0.000 | 0.000 | 1.000 | 0.000 | 0.000 |
| 19 | 1.000 | 0.000 | 0.000 | 1.000 | 0.000 | 0.000 |
| 20 | 1.000 | 0.000 | 0.000 | 1.000 | 0.000 | 0.000 |

of dimensions seems not only to be correct, but also has a significantly lower total error than GNP with the maximal number of dimensions for each number of landmarks.

We also performed the same tests to compare the performance of *OPTDIM-SP-R* with *OPTDIM-SP*. To our surprise, the tests showed, that *OPTDIM-SP-R* has significantly larger errors than *OPTDIM-SP*. Further investigations have shown, that the error difference is very small for each sample. The median of the RTT prediction error between *OPTDIM-SP* and *OPTDIM-SP-R* was less than $1\%$ of the predicted distance. The value distribution of the prediction error was very asymmetric – most of the values were in the very low range (less than $1.5\%$) and we only had a few outliers with very high values. We assume, that there are two reasons for this result. The first reason is the numerical precision loss of the Simplex Downhill function minimization due to the reduced number of variables, which explains most of the small error values. The reason for the outlier with high errors is the increased probability that the Simplex Downhill will be stuck in a local minimum when the number of possible solutions is reduced. Depending on the start point for the function minimization (initial simplex), the Simplex Downhill method may converge to a local minimum, which must not necessarily be the global minimum. In the case where the number of solutions is not reduced, there is a higher probability that the Simplex Downhill method will find a better solution in the vicinity of the non-optimal solution and converge towards this solution.

## *C. Processing Performance*

Our accuracy analysis indicates that our error prediction is not worse than the best case of the GNP landmark positioning. Now we are interested whether our landmark positioning has any performance gains compared to the standard GNP positioning. To verify this, we have compared the execution time (in terms of CPU time spent by the execution process) for the following positionings: *MAX*, *OPTDIM*, *OPTDIM-SP* and *OPTDIM-SP-R.*

If our assumptions are right, the execution time of *MAX*

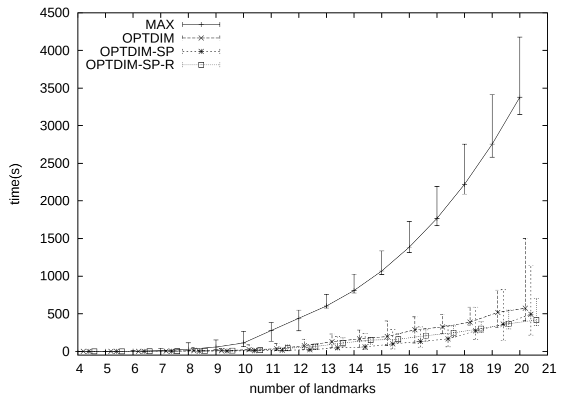

Fig. 5. Performance comparison for the plain GNP

should be the worst of all alternatives followed by *OPTDIM*, *OPTDIM-SP* should be faster than *OPTDIM* but slower than *OPTDIM-SP-R* and *OPTDIM-SP-R* should be the fastest.

We represent the experiment's results (the execution time in Fig. 5 and 6 and number of dimensions in Fig. 7) using plots. The plots summarize the results of 20 experiments grouped for each number of landmarks using bars. The middle point of a bar represents the median of all values. The lower point of a bar represents the minimum of all values. The upper point

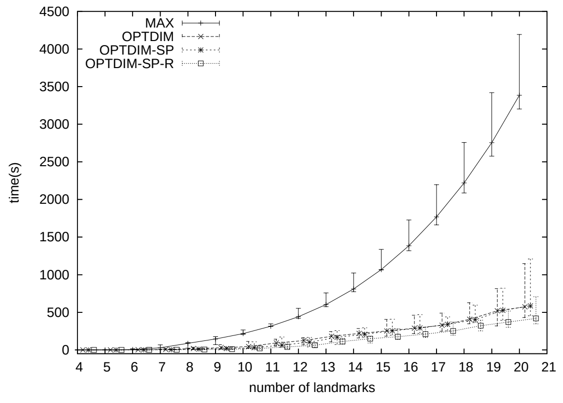

Fig. 6. Performance comparison for the normalized GNP

of a bar represents the maximum. The medians of the bars for the different experiments are connected with a line.

As we can see in Fig. 6 our assumption is correct for the normalized GNP but Fig. 5 shows that the performance of *OPTDIM-SP* is almost always better than the performance of *OPTDIM-SP-R*. In our opinion, the reason for this is the use of the simplex-downhill method for function minimization. At this time we cannot support this assumption with any experimental data, but we will focus our future work on replacing Simplex Downhill with some other function minimization strategy and comparing the results.

For the sake of completeness we also present in Fig. 7 the optimal number of dimensions calculated by our algorithm for the experiments.

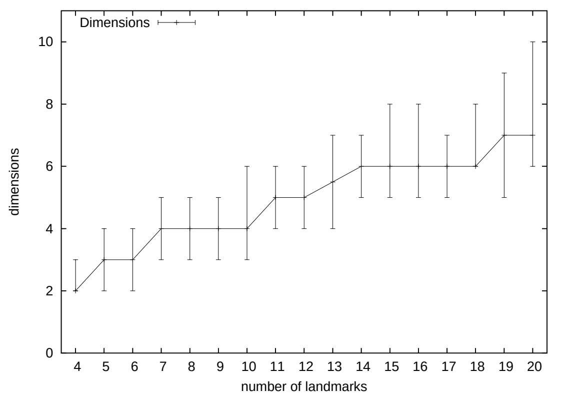

Fig. 7. Dimensions for Number of landmarks

## VIII. Conclusions

In this paper we showed that it is possible to determine the optimal number dimensions $d$ for embedding landmarks in a Euclidean space. We also showed that it is possible to find a good starting point for the function minimization. Based on this, we have defined two algorithms. The first algorithm is able to find the optimal number of dimensions $d$ for embedding landmarks in a Euclidean space and all sets of landmarks that can be used as a base for that embedding. The second algorithm finds a good starting point for the function minimization used in GNP to find the landmark positions. We have validated our algorithms by using data obtained from PlanetLab "all-sites-ping" experiments. The results of our evaluation show that both algorithms we are proposing, are correct and able to accelerate the computation of the landmark positions of GNP. Given the complexity of landmark positioning, one could consider using more than one CPU for solving the task. Unfortunately we see no way how the task of iterative function minimization using the Downhill Simplex method could be parallelized, since every iteration depends on the result of the previous iteration. One task that could be parallelized is computing the objective function, which could lead to some performance gains in the cases where the overhead of inter-process communication (IPC) is lower than the CPU time needed to calculate the objective function.

## IX. Acknowledgments

This work was carried out within the Swiss National Science Foundation project "Efficient and Robust Overlay Networks (ERON)" (SNF No. 200021-109270/1).

## References

[1] S. Floyd and K. Fall, "Router mechanisms to support end-to-end congestion control," Lawrence Berkeley National Laboratory, Tech. Rep., 1997.
[2] Y. Shavitt and T. Tankel, "Big-bang simulation for embedding network distances in euclidean space," *IEEE/ACM Trans. Netw.*, vol. 12, no. 6, pp. 993–1006, 2004.
[3] T. S. E. Ng and H. Zhang, "Predicting internet network distance with coordiantes-based approaches," in *IEEE Infocom02*, New York / USA, June 23-27 2002.
[4] ——, "A network positioning system for the internet," in *USENIX 2004*, Boston MA, USA, June 2004, pp. 141–154.
[5] M. Costa, M. Castro, A. Rowstron, and P. Key, "Pic: Practical internet coordinates for distance estimation," in *International Conference on Distributed Systems*, Tokyo, Japan, March 2004.
[6] F. Dabek, R. Cox, F. Kaashoek, and R. Morris, "Vivaldi: a decentralized network coordinate system," in *SIGCOMM '04: Proceedings of the 2004 conference on Applications, technologies, architectures, and protocols for computer communications*. New York, NY, USA: ACM Press, 2004, pp. 15–26.
[7] C. de Launois, "A stable and distributed network coordinate systems," Université catholique de Louvain, Tech. Rep., December 2004.
[8] H. Lim, J. C. Hou, and C.-H. Choi, "Constructing internet coordinate system based on delay measurement," in *Internet Measurement Confgerence 03*, October 2003.
[9] L. Tang and M. Crovella, "Virtual landmarks for the internet," in *Internet Measurement Confgerence 03*, October 2003.
[10] Y. Shavitt and T. Tankel, "Predicting internet network distance with coordiantes-based approaches," in *INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies*, Hong Kong / PRC, March 7-11 2004, pp. 374–384.
[11] J. Nelder and R. Mead, "A simplex method for function minimization," *Computer Journal*, vol. 7, pp. 308–313, 1965.
[12] C. Yoshikawa. Planetlab all-sites-pings experiment. [Online]. Available: http://ping.ececs.uc.edu/ping/
[13] L. M. Blumenthal, *Theory and Applications of Distance Geometry*. Oxford: Clarendon Press, 1953.
[14] D. M. Y. Sommerville, *An Introduction to the Geometry of n Dimensions*. New York: Dover Publications, 1958, p. 124.
[15] J. Hietaniemi. Comprehensive perl archive network. [Online]. Available: http://www.cpan.org/
[16] F. Wilcoxon, "Individual comparisons by ranking methods," *Biometrics*, vol. 1, pp. 80–83, 1945.
[17] J. W. Eaton. Octave. [Online]. Available: http://www.gnu.org/software/octave/