Network Distance Prediction Based on Decentralized Matrix Factorization

Yongjun Liao, Pierre Geurts, Guy Leduc

Network Distance Prediction Based on Decentralized Matrix Factorization

Yongjun Liao[1], Pierre Geurts[2,3], Guy Leduc[1]

[1] Research Unit in Networking (RUN), University of Liège, Belgium
{liao,leduc}@run.montefiore.ulg.ac.be
[2] Systems and Modeling, University of Liège, Belgium
[3] Research associate, FRS-F.N.R.S. (Belgium)
p.geurts@ulg.ac.be

Abstract. Network Coordinate Systems (NCS) are promising techniques to predict unknown network distances from a limited number of measurements. Most NCS algorithms are based on metric space embedding and suffer from the inability to represent distance asymmetries and Triangle Inequality Violations (TIVs). To overcome these drawbacks, we formulate the problem of network distance prediction as guessing the missing elements of a distance matrix and solve it by matrix factorization. A distinct feature of our approach, called Decentralized Matrix Factorization (DMF), is that it is fully decentralized. The factorization of the incomplete distance matrix is collaboratively and iteratively done at all nodes with each node retrieving only a small number of distance measurements. There are no special nodes such as landmarks nor a central node where the distance measurements are collected and stored. We compare DMF with two popular NCS algorithms: Vivaldi and IDES. The former is based on metric space embedding, while the latter is also based on matrix factorization but uses landmarks. Experimental results show that DMF achieves competitive accuracy with the double advantage of having no landmarks and of being able to represent distance asymmetries and TIVs.

Key words: Network Coordinate System, Matrix Factorization, Decentralized Matrix Factorization, Regularization

1 Introduction

Predicting network distances (e.g. delay) between Internet nodes is beneficial to many Internet applications, such as overlay routing [1], peer-to-peer file sharing [2], etc. One promising approach is Network Coordinate Systems (NCS), which construct models to predict the unmeasured network distances from a limited number of observed measurements [3].

Most NCS algorithms embed network nodes into a metric space such as Euclidean coordinate systems in which distances between nodes can be directly computed from their coordinates. For example, GNP [4] is the first system that models the Internet as a geometric space. It first embeds a number of landmarks into the space and a non-landmark host determines its coordinates with respect to the landmarks. Vivaldi [5] is a decentralized NCS system that extends GNP by eliminating the landmarks. It simulates a system of springs where each edge is modeled by a spring and the force of the spring reflects the approximation error.

However, network distances do not derive from the measurements of a metric space. For example, Triangle Inequality Violations (TIVs) have been frequently observed and the distances between two nodes are not necessarily symmetric due to the network structure and routing policy [6–8, 5, 9]. No algorithm based on metric space embedding can model such distance space. IDES [10] is one of the few algorithms using non-metric space embedding techniques. It is based on matrix factorization which approximates a large matrix by the product of two small matrices. A drawback of IDES is that, similar to GNP, IDES also relies on landmarks. It factorizes a small distance matrix built from the landkmarks at a so-called information server and other non-landmark nodes compute their coordinates with respect to the landmarks. Phoenix extends IDES by adopting a weight model and a non-negativity constraint in the factorization to enforce the predicted distances to be positive [11]. In practice, NCS systems with landmarks are less appealing. They suffer from landmark failures and overloading. Furthermore, the number of landmarks and their placement affect the performance of NCS.

In this paper we propose a novel approach, called Decentralized Matrix Factorization (DMF), to predicting network distance. Unlike IDES, we seek to factorize a distance matrix built from all nodes in a fully decentralized manner. Each node retrieves distance measurements from and to a small number of randomly selected nodes[1] and updates its coordinates simultaneously and iteratively. There are no special nodes such as landmarks or a central node where distance measurements are collected and stored. In doing so, our DMF algorithm overcomes the drawbacks of metric space embedding and is able to represent asymmetric distances and TIVs, while it is fully decentralized and requires no landmarks. Experimental results show that DMF is stable and achieves competitive performance compared to IDES and metric space embedding based algorithms such as Vivaldi.

The rest of the paper is structured as follows. Section 2 formulates the problem of network distance prediction by matrix factorization. Section 3 describes the DMF algorithm. Section 4 presents the evaluation and the comparison of DMF with other competing methods. Section 5 gives the conclusions and discusses future work.

2 Matrix Factorization for Network Distance Prediction

Matrix Factorization seeks an approximate factorization of a large matrix, i.e.,

$$D \approx \hat{D} = XY^T,$$

[1] We will refer to the selected nodes as neighbors in the rest of the paper.

where the number of columns in X and Y is typically small and is called the dimension of the embedding space. Generally, the factorization is done by minimizing $||D - \hat{D}||^2$, which can be solved analytically by using singular value decomposition (SVD) [12]. In many cases, constraints can be imposed in the minimization. A popular and useful constraint is that elements of X and Y are non-negative. This so-called non-negative matrix factorization (NMF) can only be solved by iterative optimization methods such as gradient descent [13]. Note that matrix factorization has no unique solution as

$$D \approx \hat{D} = XY^T = XGG^{-1}Y^T,$$

where G is any arbitrary non-singular matrix. Therefore, replacing X by XG and Y by $G^{-1}Y$ will not increase the approximation error.

In using matrix factorization for network distance prediction, assuming n nodes in the network, a $n \times n$ distance matrix D is constructed with some distances between nodes measured and the others unmeasured. To guess the missing elements in D, we factorize D into the form $D \approx XY^T$ by solving

$$\mathbf{min}\ ||W. * (D - XY^T)||^2, \tag{1}$$

where $.*$ is element-wise product and W is the weight matrix with $w_{ij} = 1$ if d_{ij}, the distance from i to j, is measured and 0 otherwise. X and Y are of the same size $n \times l$ with $l \ll n$. l is referred to as the dimension of the embedding space and is a parameter of the factorization algorithm.

With missing elements, the minimization of eq. 1 can only be solved by iterative optimization methods. After the factorization, each node is then associated with two coordinates x_i and y_i, where x_i is the ith row of X, called *outgoing vector*, and y_i is the ith row of Y, called *incoming vector*. The estimated distance from i to j is

$$\hat{d}_{ij} = x_i \cdot y_j, \tag{2}$$

where $\cdot$ is the dot product. If done properly, the estimated distance, $\hat{d}_{ij}$, approximates the measured distance, d_{ij}, within a limited error range. Note that $\hat{d}_{ij}$ is not necessarily equal to $\hat{d}_{ji}$.

The above process is centralized and requires a large number of distance measurements to be collected and stored at a central node. To solve this problem, IDES [10] proposed to select a small number of landmarks and compute, at a so-called information server, the factorization (by using SVD or NMF) of a small distance matrix built only from measured distances between the landmarks. Once the landmark coordinates have been fixed, a non-landmark host can determine its coordinates by measuring its distance to and from each of the landmarks and finding coordinates that most closely match those measurements. As mentioned earlier, the use of landmarks is a weakness of IDES. In the next section, we will propose our approach based on a decentralized matrix factorization that eliminates the need for landmarks.

3 Decentralized Matrix Factorization for Network Distance Prediction

The problem is the same as in eq. 1, but we seek to solve it in a decentralized manner. Similar to IDES, each node records its outgoing vector x_i and incoming vector y_i and computes distances from and to other nodes by using eq. 2. The difference is that x_i and y_i are initialized randomly and updated continuously with respect to some randomly-selected neighbors.

In particular, to update x_i and y_i, node i randomly selects k neighbors, measures its distances from and to them, and retrieves the outgoing and incoming vectors. Denote $X_i = [x_{i_1}; \ldots; x_{i_k}]$ and $Y_i = [y_{i_1}; \ldots; y_{i_k}]$ the outgoing and incoming matrices built from the neighbors of i, i.e., x_{i_j} and y_{i_j} are the outgoing and incoming vectors of the jth neighbors of i. Let $d^i_{to} = [d_{i,i_1}, \ldots, d_{i,i_k}]$ and $d^i_{from} = [d_{i_1,i}, \ldots, d_{i_k,i}]$ the distance vectors to and from the neighbors of i. Then, x_i and y_i are updated by

$$x_i = \arg\min_x \; ||xY_i^T - d^i_{to}||^2, \tag{3}$$

$$y_i = \arg\min_y \; ||X_i y^T - {d^i_{from}}^T||^2. \tag{4}$$

Eqs. 3 and 4 are standard least square problems of the form $min||Ax - b||^2$, which has an analytic solution of the form:

$$x = (A^T A)^{-1} A^T b. \tag{5}$$

To increase the numerical stability of the solution, instead of solving eqs. 3 and 4 with eq 5, we penalize x_i and y_i and solve regularized least square problem of the form $min||Ax - b||^2 + \lambda||x||^2$, which also has an analytic solution

$$x = (A^T A + \lambda I)^{-1} A^T b, \tag{6}$$

where λ is the coefficient of the regularization term. In the experimental section, we will show the influence of the regularization terms on the performance of DMF.

To summarize, the update equations of x_i and y_i are

$$x_i = d^i_{to} Y_i (Y_i^T Y_i + \lambda I)^{-1} \tag{7}$$

$$y_i = d^i_{from} X_i (X_i^T X_i + \lambda I)^{-1} \tag{8}$$

The DMF algorithm is given in Algorithm 1[2]. We initialize the coordinates with random numbers uniformly distributed between 0 and 1. Empirically, we found that DMF is insensitive to the random initialization of the coordinates.

[2] Note that we can also adopt the weight model and the non-negativity constraint as in [11]. However, the constrained minimization in eqs 3 and 4 have no more closed form solutions and has to be solved by iterative optimization methods. As claimed in [10], which is confirmed by our experiments, the non-negativity constraint does not improve the accuracy a lot, but significantly increases the computing time.

Input: D, l, k, λ
D: distance matrix with missing elements
l: dimension of the embedding space
k: number of neighbors of each node
λ: regularization coefficient
Output: X,Y
foreach *node i* **do**
    Randomly select k neighbors from the network.
    Randomly initialize x_i and y_i.
    while *forever* **do**
        retrieve d^i_{to}, d^i_{from}, X_i, Y_i;
        update x_i by eq. 7
        update y_i by eq. 8
        sleep some time
    end
end

Algorithm 1: DMF: Decentralized Matrix Factorization with Regularization.

4 Experiments and Evaluations

In this section, we evaluate DMF[3] and compare it with two popular NCS algorithms: Vivaldi and IDES. The former is based on metric space embedding, while the latter is also based on matrix factorization but uses landmarks. All the experiments are performed on two typical data sets collecting real Internet measurements: the P2psim [14] data set which contains the measured distances between 1740 Internet DNS servers, and the Meridian [15] data set which contains the measured distances between 2500 nodes. While DMF can in principle handle asymmetric distance matrices, in our experiment, we took $d_{i,j} = d_{j,i}$ and defined these distances as the half of the round-trip-time between nodes i and j. The same assumption is adopted in Vivaldi and has the advantage of greatly simplifying the implementation of the algorithm, as measuring one-way delay is difficult in practice.

In the simulations, we randomly selected a node and updated its coordinates at each step. An iteration of a simulation is defined by a fixed round of node updates. Since Vivaldi updates its coordinates with respect to only one neighbor in contrast to DMF that does it with respect to all neighbors, an iteration in Vivaldi is defined by $n \times k$ node updates whereas in DMF an iteration is n node updates, where n is the number of nodes and k is the number of neighbors. In doing so, we ensure that, on average, all nodes have a chance to update their coordinates with respect to all neighbors. Note that IDES is not an iterative method. The coordinates of the nodes are unchanged.

We examine the following classical evaluation criteria.

[3] A matlab implementation of DMF used to generate the results in the paper can be downloaded from `http://www.run.montefiore.ulg.ac.be/~liao/DMF`.

– *Cumulative Distribution of Relative Estimation Error* Relative Estimation Error (REE) is defined as

$$REE = \frac{|\hat{d}_{i,j} - d_{i,j}|}{d_{i,j}}.$$

– *Stress* measuring the overall fitness of the embedding is defined as

$$stress = \sqrt{\frac{\sum_{i,j}(d_{i,j} - \hat{d}_{i,j})^2}{\sum_{i,j} {d_{i,j}}^2}}.$$

– *Median Absolute Estimation Error* (MAEE) is defined as

$$MAEE = median_{i,j}(|d_{i,j} - \hat{d}_{i,j}|).$$

Note that our DMF algorithm utilizes only a small percentage of the distance measurements in the datasets to estimate the coordinates of the nodes, but the evaluation of the above criteria is done using all distance measurements.

4.1 Parameter Tuning

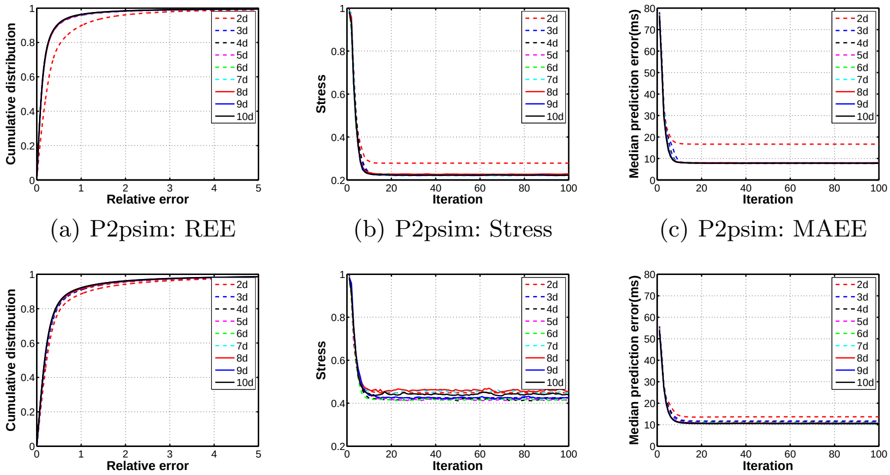
DMF has three parameters to be defined: l, the dimension of the embedding space, k, the number of neighbors of each node, and λ, the regularization coefficient. We study the influence of these parameters on the performance of DMF. To this end, we tune one parameter at a time while fixing the other two. Results are shown in Figures 1, 2 and 3.

It can be seen that l does not seem to affect the performance of DMF as long as $l \geq 3$ which coincides with the conclusion drawn in [5] about Vivaldi. We nevertheless recommend $l = 10$ as it does not pose any problem and as the same number is used in IDES. On the other hand, k has a clear impact, as a larger k gives better accuracy, which is obvious because a larger k means fewer missing elements thus better estimation of the coordinates. However, a larger k also means more probe traffic and a higher overhead. Following Vivaldi, we suggest $k = 32$ as a good tradeoff between accuracy and measurement overhead. For λ, too little or too much regularization only decreases the accuracy of DMF, and 50 seems to be a good choice for both P2psim and Meridian datasets.

Note that the results are very stable from one simulation to another, as highlighted in Figure 4. The algorithm does not seem very sensitive to the random initialization of the coordinates and to the particular selection of neighbours. In the following, unless otherwise stated, $l = 10$, $k = 32$ and $\lambda = 50$ are used by default and the results of all the experiments are derived from one simulation.

4.2 Analysis of Convergence and Stability

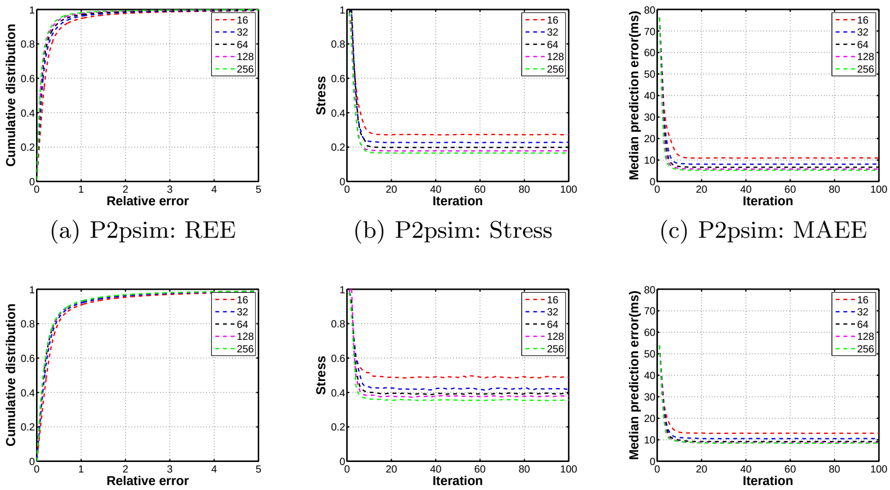
We further evaluate the convergence and the stability of DMF. From Figures 1, 2, 3 and 4, it is clear that DMF converges fast, empirically in less than 20 iterations for both P2psim and Meridian datasets.



(a) P2psim: REE (b) P2psim: Stress (c) P2psim: MAEE

(d) Meridian: REE (e) Meridian: Stress (f) Meridian: MAEE

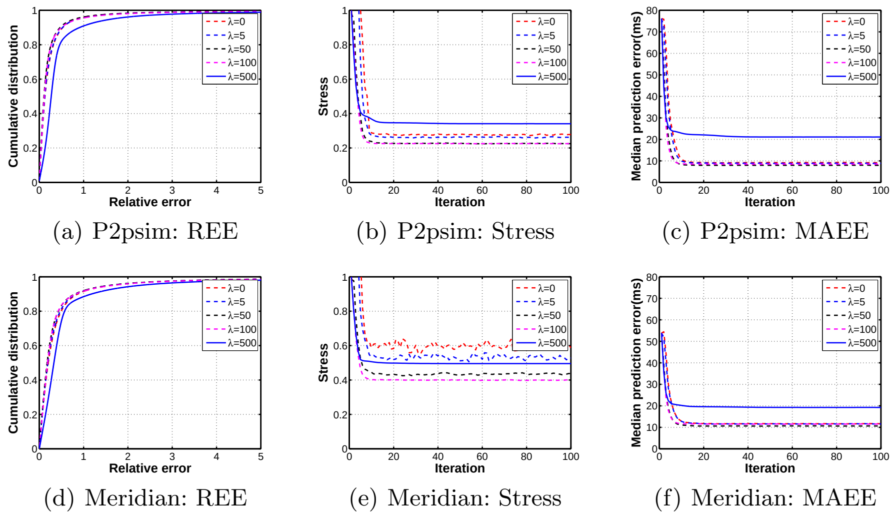
Fig. 1. The effect of the dimension (l) on the performance of DMF. ($k = 32$, $\lambda = 50$)



(a) P2psim: REE (b) P2psim: Stress (c) P2psim: MAEE

(d) Meridian: REE (e) Meridian: Stress (f) Meridian: MAEE

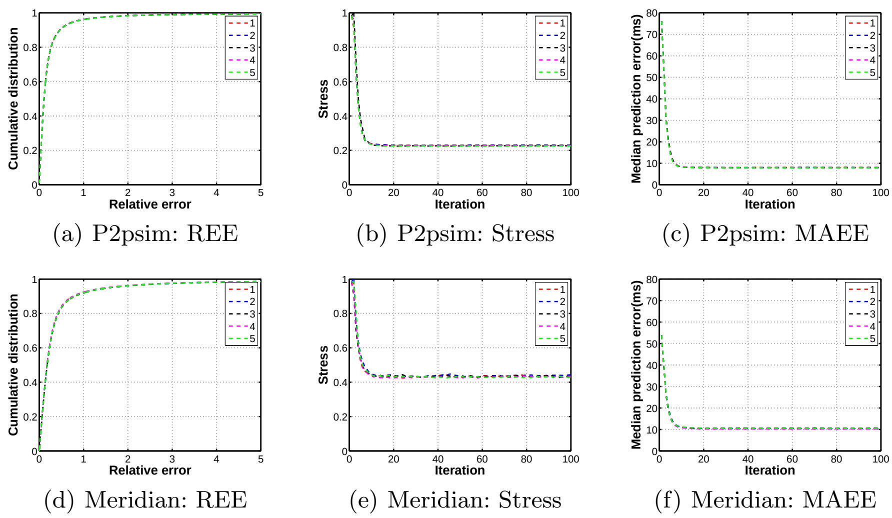
Fig. 2. The effect of the number of neighbors (k) on the performance of DMF. ($l = 10$, $\lambda = 50$)



(a) P2psim: REE (b) P2psim: Stress (c) P2psim: MAEE

(d) Meridian: REE (e) Meridian: Stress (f) Meridian: MAEE

Fig. 3. The effect of regularization coefficient (λ) on the performance of DMF. ($l = 10$, $k = 32$)



(a) P2psim: REE (b) P2psim: Stress (c) P2psim: MAEE

(d) Meridian: REE (e) Meridian: Stress (f) Meridian: MAEE

Fig. 4. Results of different simulations. The simulations differ in the initializations of the coordinates, in the selections of the neighbors by each node and in the orders in which the nodes are updated. It can be seen that the results are insensitive to these differences.

To further verify the stability of DMF, we performed a 2D factorization ($l = 2$) and plotted the X and Y coordinates at different times of the simulation, shown in Figure 5. It can be seen that the coordinates are very stable with little drift after the embedding errors become stable. Figure 6 shows the histogram of the differences between the predicted distance matrix at the 20th and the 100th iterations.

4.3 Comparisons with Vivaldi and IDES

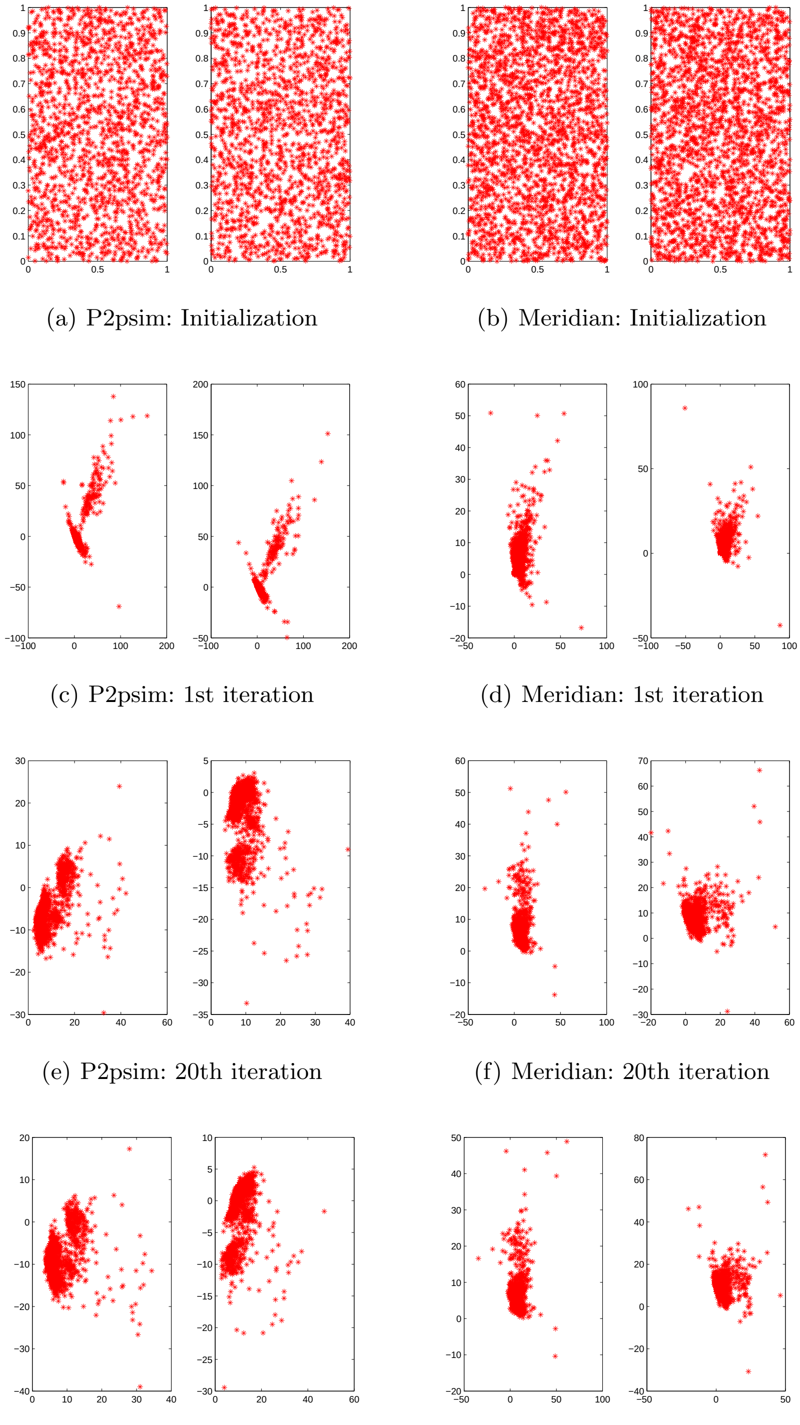
Lastly, we compare DMF with Vivaldi and IDES, as shown in Figure 7. Vivaldi is a decentralized NCS algorithm based on Euclidian embedding. Similar to DMF, each node updates its coordinates with respect to k randomly selected neighbors. Here, we took $k = 32$ following the recommendation in Vivaldi. For IDES, a number of landmarks are needed. Although [16, 10] claimed that 20 randomly selected landmarks are sufficient to achieve desirable accuracy, we nevertheless deployed 32 landmarks in our experiments for the purpose of comparison. The dimensions of the embedding space are 10 for all algorithms.

From Figure 7, it can be seen that both DMF and Vivaldi achieve similar accuracy and slightly outperform IDES. The worse performance by IDES is likely due to the use of the landmarks. Since in IDES, a non-landmark node only communicates with landmarks, no links between non-landmark nodes are used by the NCS. In contrast, DMF and Vivaldi are completely decentralized with links between nodes randomly selected and evenly distributed in the whole network.

5 Conclusions and Future Works

In this paper, we proposed a novel approach, called DMF, to predicting unknown network distances. Essentially, we consider it as a learning problem where coordinates of network nodes are learned from partially observed measurements and the unknown distances are approximated from the learned coordinates. Different from all previous works, the learning of the coordinates is done by DMF which requires no landmarks. Since DMF is not based on metric space embedding, it has the potential to overcome common limitations such as the inability to represent TIVs and asymmetric distances. Experimental results show that the performance of our approach is comparable with two popular NCS algorithms: Vivaldi and IDES.

The reason why DMF does not outperform Vivaldi given the above-mentioned advantages remains an open question that needs further investigation. It has to be noted that both P2psim and Meridian datasets are symmetric with $d_{ij} = d_{ji}$. We would like to test DMF on more datasets, especially those with heavily asymmetric distances.



(a) P2psim: Initialization

(b) Meridian: Initialization

(c) P2psim: 1st iteration

(d) Meridian: 1st iteration

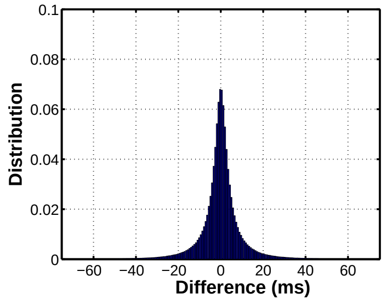
(e) P2psim: 20th iteration

(f) Meridian: 20th iteration

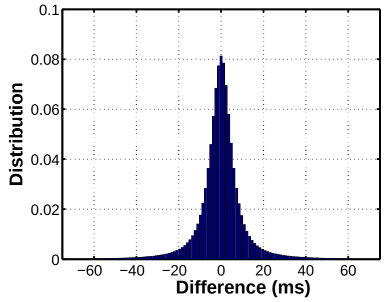
(g) P2psim: 100th iteration

(h) Meridian: 100th iteration

Fig. 5. The evolution of the coordinates, X(left subplot) and Y(right subplot).

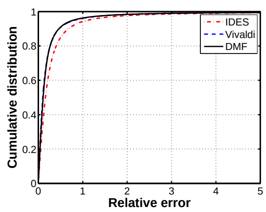


(a) P2psim: difference

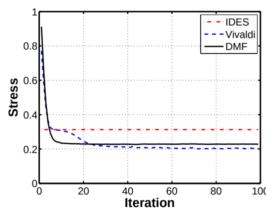


(b) Meridian: difference

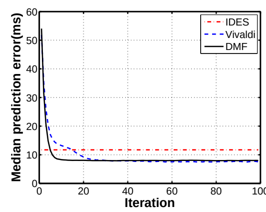
Fig. 6. The differences between the predicted distance matrix at the 20th and the 100th iterations.



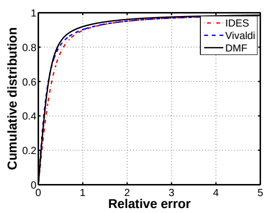
(a) P2psim: REE



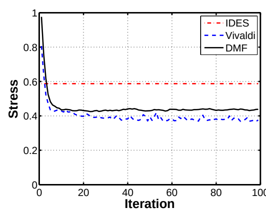
(b) P2psim: Stress



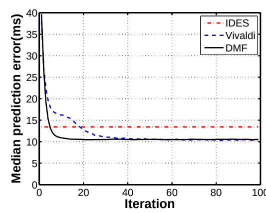
(c) P2psim: MAEE



(d) Meridian: REE



(e) Meridian: Stress



(f) Meridian: MAEE

Fig. 7. Comparison with IDES and Vivaldi. ($l = 10$, $k = 32$, $\lambda = 50$ for DMF)

Acknowledgements

This work has been partially supported by the EU under projects FP7-Fire ECODE, by the European Network of Excellence PASCAL2 and by the Belgian network DYSCO (Dynamical Systems, Control, and Optimization), funded by the Interuniversity Attraction Poles Programme, initiated by the Belgian State, Science Policy Office. The scientific responsibility rests with its authors.

References

1. Hari, D.A., Andersen, D., Balakrishnan, H., Kaashoek, F., Morris, R.: Resilient overlay networks. (2001) 131–145
2. Azureus Bittorrent. `http://azureus.sourceforge.net`.
3. Donnet, B., Gueye, B., Kaafar, M.A.: A survey on network coordinates systems, design, and security. To appear in IEEE Communication Surveys and Tutorial (Dec 2010)
4. Ng, T.S.E., Zhang, H.: Predicting Internet network distance with coordinates-based approaches. In: Proc. IEEE INFOCOM, New York, NY, USA (June 2002)
5. Dabek, F., Cox, R., Kaashoek, F., Morris, R.: Vivaldi: A decentralized network coordinate system. In: Proc. ACM SIGCOMM, Portland, OR, USA (August 2004)
6. Zheng, H., Lua, E.K., Pias, M., Griffin, T.: Internet Routing Policies and Round-Trip-Times. In: Proc. the PAM Conference, Boston, MA, USA (April 2005)
7. Lee, S., Zhang, Z., Sahu, S., Saha, D.: On suitability of euclidean embedding of internet hosts. SIGMETRICS **34**(1) (2006) 157–168
8. Wang, G., Zhang, B., Ng, T.S.E.: Towards network triangle inequality violation aware distributed systems. In: Proc. the ACM/IMC Conference, San Diego, CA, USA (oct 2007) 175–188
9. Banerjee, S., Griffin, T.G., Pias, M.: The interdomain connectivity of PlanetLab nodes. In: Proc. of the Passive and Active Measurement Workshop – PAM'2004. Lecture Notes in Computer Science (LNCS) 3015, Antibes Juan-les-Pins, France (April 2004)
10. Mao, Y., Saul, L., Smith, J.M.: Ides: An internet distance estimation service for large networks. IEEE Journal On Selected Areas in Communications (JSAC), Special Issue on Sampling the Internet, Techniques and Applications **24**(12) (Dec 2006) 2273–2284
11. Chen, Y., Wang, X., Song, X., Lua, E.K., Shi, C., Zhao, X., Deng, B., Li., X.: Phoenix: Towards an accurate, practical and decentralized network coordinate system. In: Proc. IFIP Networking Conference, Aachen, Germany (May 2009)
12. Golub, G.H., Van Loan, C.F.: Matrix computations (3rd ed.). Johns Hopkins University Press, Baltimore, MD, USA (1996)
13. Lee, D.D., Seung, H.S.: Algorithms for non-negative matrix factorization. In: NIPS, MIT Press (2001) 556–562
14. A simulator for peer-to-peer protocols. `http://www.pdos.lcs.mit.edu/p2psim/index.html`.
15. Wong, B., Slivkins, A., Sirer, E.: Meridian: A lightweight network location service without virtual coordinates. In: Proc. the ACM SIGCOMM. (aug 2005)
16. Tang, L., Crovella, M.: Geometric exploration of the landmark selection problem. In: Proc. of the Passive and Active Measurement Workshop – PAM'2004. Lecture Notes in Computer Science (LNCS) 3015, Antibes Juan-les-Pins, France (April 2004)