

Myth: An Accurate and Scalable Network Coordinate System under High Node Churn Rate

Yang Chen^{1,2}, Genyi Zhao³, Ang Li⁴, Beixing Deng¹, Xing Li¹

¹Department of Electronic Engineering, Tsinghua University, China

²Department of Computer Science, Stanford University

³Department of Automation, Tsinghua University, China

⁴School of Information and Computer Science, University of California, Irvine

Abstract—Network coordinate (NC) system is an efficient mechanism to predict Internet distance (latency) with limited times of measurement. In this paper, we focus on the node churn problem in distributed NC systems. Our study on Vivaldi, a representative distributed NC system, shows that under high node churn rate the prediction accuracy of the system will be seriously impaired, which renders the system unusable in many potential application scenarios. As remedy, we propose the idea and implementation of Myth, an accurate and scalable NC system even under high node churn rate. Myth introduces the merit of Landmark-based NC system to shorten convergence time in Simulation-based NC system with negligible extra overhead. We evaluate the performance of Myth system with two typical real-world data sets measured in Internet. The experimental results show that Myth outperforms Vivaldi in relative error, relative rank loss and closest neighbor absolute error under high node churn rate, without compromising the performance under stable environment.

Index Terms—Peer-to-Peer, Network Coordinates, Node Churn, Accuracy, Scalability

I. INTRODUCTION

Network coordinate (NC) system is an efficient mechanism to predict network distance (latency) between any two Internet nodes without explicit measurements. In NC system, each node is assigned a set of numbers called network coordinates to represent its position in an Euclidean space, and the latency between any two nodes can be calculated from their coordinates with a distance function. NC system reduces the active probing overhead significantly, which is especially beneficial to large-scale distributed applications, such as peer-to-peer content distribution, multi-user gaming, and server selection.

Several approaches have been proposed in literature, whose core algorithms can be categorized into two classes, namely landmark-based algorithm (LBA) and simulation-based algorithm (SBA). In LBA systems ([2], [7], [8]), each node measures its distances to a set of reference nodes called landmarks, and its coordinates can be determined by minimizing the difference between the measured distances and the calculated distances. LBA provides high accuracy and stability. However, the dedicated landmark nodes are under the heavy load of serving all the other nodes in the system. Also, as the common weakness of all centralized systems, the failure of landmark nodes will degrade the prediction accuracy of the whole NC system, which introduces single point of failure to the system. As for SBA systems, such as [3] and [14], pairwise distances

are mapped into a physical spring system, so that every node's coordinates can be determined by minimizing the energy of the whole simulated physical system. SBA systems distribute the measurement and computation to all participating nodes in order to lighten the load of any individual node. Thus SBA guarantees higher scalability and has received more practical implementations.

In Vivaldi [3], a representative SBA system, each node starts from the same initial position in the Euclidean space, and takes many rounds to update its own coordinates before converging to the ideal position, where energy of the whole system is the lowest. Each update tries to minimize the energy by referring to the measured distance to one of its neighbors and the neighbor's current coordinates. Given that the convergence time of a Vivaldi node will take tens of seconds even when nodes stay stable in the system, it is very likely that one node's convergence process be interfered by its neighbors' leaving and joining, which are common in general P2P systems. [1] indicates that almost every distributed system has to deal with *churn*-change in the set of participating nodes due to joins, graceful leaves, and failures. Moreover, the newly joined nodes with inaccurate coordinates will be referred by other nodes in the hope to improve their own coordinates, which increases the instability of the whole system. Thus, the lengthy convergence time, together with the inaccurate initial positions of nodes, might significantly impair Vivaldi's prediction accuracy in high node churn rate scenario. [5] confirms this by analyzing the statistics obtained from Azureus BitTorrent client running upon Vivaldi.

This paper focuses on the node churn problem of Vivaldi, with the major contributions as follows:

First, we study the impact of the node churn to the Vivaldi system. By evaluating the performance of Vivaldi under high node churn rate we can see that the system will have degraded prediction accuracy under such situation.

Second, we propose Myth, a fully decentralized NC scheme, to solve this problem. In Myth, a dedicated LBA-like algorithm is designed to intelligently choose the initial coordinates of nodes based on existing information. After the initial coordinates are determined, SBA is utilized to achieve convergence and guarantee scalability. By using the algorithm, the initial position of a Myth node is close to its final ideal position; hence the convergence time is greatly shortened. Furthermore,

because the initial coordinates of a Myth node are more accurate than the ones of a Vivaldi node, the impairment of referring to a newly joined node as neighbor is reduced. The extra overhead of using Myth is moderate. We evaluate the performance of Myth by comparing it with the Vivaldi system. The experimental results show that under high node churn rate Myth outperforms Vivaldi under every metric. Moreover, in both stable and churn scenarios, Myth achieves almost the same prediction accuracy. We believe that Myth is a step further towards a practical NC system in high node churn rate environment.

The rest of this paper is organized as follows. In Section II, we review the related works. Then in Section III, we present the design and implementation of Myth, followed by its performance evaluation in Section IV. In Section V, we summarize the conclusions.

II. IMPACT OF HIGH NODE CHURN RATE FOR VIVALDI

In this section, we first briefly introduce the basic algorithm of Vivaldi, and then study its performance degradation under high node churn rate.

A. Vivaldi Algorithm

Vivaldi characterizes the whole network as a spring system. Let L_{ij} be the actual distance (round-trip time) between node i and node j in the system, and x_i be the coordinates assigned to node i . Ideally, the coordinates of all nodes are produced by minimizing the following error function which corresponds to the overall energy. We refer to those coordinates as ideal coordinates of all nodes.

$$E = \sum_i \sum_j (L_{ij} - \|x_i - x_j\|)^2 \quad (1)$$

$\|x_i - x_j\|$ is the distance between node i and node j calculated based on coordinates.

In practical decentralized Vivaldi system, node i maintains its current coordinates x_i and local error e_i , and updates them many rounds to get closer to the ideal coordinates. In each round, i adjusts them through measuring the latency to one of its overlay neighbor in the system. The pseudo-code of Vivaldi is shown as follows, where c_e and c_c are tunable parameters.

Algorithm 1 *vivaldi*(r_{tt}, x_j, e_j)

- 1: $w = e_i / (e_i + e_j)$
 - 2: $e_s = \| \|x_i - x_j\| - r_{tt} \| / r_{tt}$
 - 3: $e_i = e_s \times c_e \times w + e_i \times (1 - c_e \times w)$
 - 4: $\delta = c_c \times w$
 - 5: $x_i = x_i + \delta \times (r_{tt} - \|x_i - x_j\|) \times u(x_i - x_j)$
-

A sample weight is firstly computed based on the local and remote error (line 1), and then the relative error is computed (line 2). Next node i updates its local error (line 3). Finally node i calculates and updates its coordinates (line 4 and line 5).

In Vivaldi, all nodes have the same initial coordinates. To bootstrap the algorithm Vivaldi defines $u(0)$ as a unit-length vector in a randomly chosen direction. When two nodes occupy the same location, there will be a spring pushing them away from each other in an arbitrary direction.

B. Impact of High Node Churn Rate

In most of the NC applications, especially peer-to-peer content distribution or file sharing, all nodes may join or leave the system at any time. This inherent dynamics and distributed essence make the deployment of NC even more challenging.

[3] only studied Vivaldi under stable environment where all nodes will not leave the system after they join. However, this is not always the case in practice. [5] shows that the client lifetime in file sharing using Azureus (running upon Vivaldi) follows a long-tailed distribution typical in P2P systems. In their study, 78% of the nodes stay in the system for less than 60 minutes.

We evaluate Vivaldi in two scenarios and compare the prediction accuracies.

Stable scenario: All nodes join the system at the beginning and stay throughout the whole experiment.

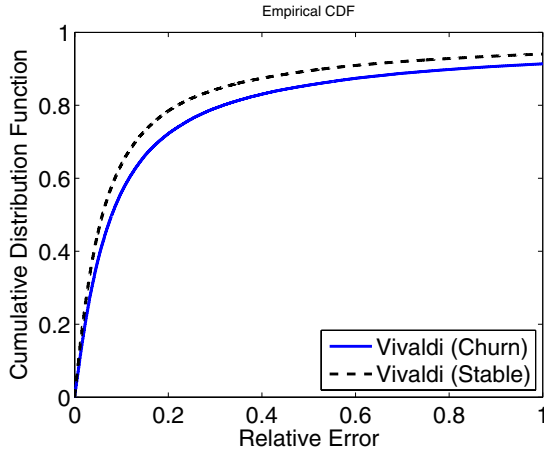
Churn scenario: Each node joins at the beginning and stays for a random period of time characterized by the Pareto Distribution. After a node leaves, it sleeps for a random period with uniform distribution and rejoins the system as a newcomer to stay another Pareto distributed period. Pareto Distribution can describe the long-tailed characteristic of nodes' staying time, which is also used for simulating the node staying time in [1] and [6].

In particular, the Pareto distribution is determined by parameters k and x_m (k is a shape parameter that determines how skew the distribution is, and x_m is a location parameter that determines where the distribution starts). [6] studies the cumulative distributions of the traces of PPLive, the largest commercial P2P live streaming system to date with over 100,000 users online at peak times. Their finding shows that the nodes' stay duration of PPLive is well approximated by a Pareto distribution of $k = 1.03$. Thus, in our simulation k is set as 1.03. We also adjust x_m to let the stay duration at 78th percentile to be 60 minutes.

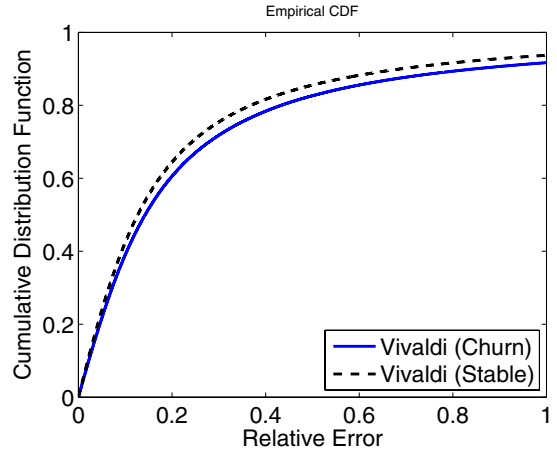
The prediction accuracy of a NC scheme is often denoted by the relative error (RE) of predicted distance over the real latency measured on Internet. Relative Error (RE) is defined as:

$$RE = \frac{|PredictionDist - MeasuredDist|}{\min(PredictionDist, MeasuredDist)} \quad (2)$$

We use two different data sets from real Internet measurement to study the prediction accuracy of Vivaldi. The first data set is the King data set from [3], which includes the round-trip latencies among 1740 Internet naming servers. The second data set is the PlanetLab data set, available at [13], which includes the round-trip latencies among 226 hosts of the Planet-Lab. As in [4], each Vivaldi node measures a round-trip time (RTT) to one of its neighbor nodes in the overlay once every five seconds.



(a) PlanetLab



(b) King

Fig. 1. Distribution of Relative Error



Fig. 2. Illustrations of Coordinates Update in Vivaldi and Myth

Fig.1 shows the comparison of the REs of Vivaldi between two different scenarios. According to Fig.1, we can find that the performance of Vivaldi degrades in node churn scenario. In *Churn scenario* ninety-percentile REs are 0.820 in PlanetLab data set and 0.848 in King data set, whereas in *Stable scenario*, ninety-percentile REs are 0.533 in PlanetLab data set and 0.689 in King data set. Vivaldi's performance degradation in node churn scenario will seriously limit its application.

III. MYTH SYSTEM DESIGN

A. Basic Idea of Myth

In Vivaldi, all the nodes start at the same initial position and update their coordinates for many rounds and finally converge to their ideal positions. Fig. 2(a) shows the first three rounds of update of two nodes after their joining to the system. The two stars with different colors show the ideal positions of both nodes, respectively.

To use the same initial coordinates for all nodes is a bad heuristic to bootstrap the system, even though nodes will be spread in the following update. First, extra rounds are needed to spread out the nodes from their initial coordinates, and hence the convergence time is increased. Furthermore, since a newly joined node may serve as the reference for its neighbors, its irrelevant and inaccurate initial coordinates will affect the prediction of its neighbors. Thus under high node churn rate, a

large portion of the nodes are in the state of disturbance with inaccurate NCs.

To remedy this problem, in Myth we manage to bootstrap each node with the initial coordinates closed to its ideal coordinates. Fig. 2(b) shows the case. An Initial Coordinates (IC) prediction scheme is added before the Vivaldi-like adjustment. The scheme utilizes the coordinates of nodes which are already in the system. After doing this, each node needs fewer rounds to converge. Moreover, the prediction error caused by referring to a newly joined node is reduced, which improves the stability of the system. In the following we will present the scheme in details.

B. Initial Coordinates Prediction

One sparking point of Myth is to predict the IC of each node in a scalable fashion. When a new node A joins the overlay system, it will create neighbor connections with nodes already in the overlay through Gossip [15]. After getting enough neighbors, node A measures its RTTs to L nodes in its neighbor set ($L > N$, where N is the dimension of the space). Using the L measured distances, node A can calculate its own initial coordinates IC_A which minimize the overall error between the measured and the calculated distances from A to these L neighbors. As in [2], we use Simplex Downhill Algorithm to minimize the following objective function $f_{obj}(A)$.

$$f_{obj}(A) = \sum_{N_i \in N_1 \dots N_L} \xi(d_{AN_i}, \hat{d}_{AN_i}) \quad (3)$$

where $\xi(\cdot)$ is an error measurement function defined as follows.

$$\xi(d_{N_1 N_2}, \hat{d}_{N_1 N_2}) = \left(\frac{d_{N_1 N_2} - \hat{d}_{N_1 N_2}}{d_{N_1 N_2}} \right)^2 \quad (4)$$

The idea of our IC prediction algorithm is similar to the NC calculation of the ordinary GNP nodes in [2]. However, there is a significant difference between Myth and GNP or other LBAs: Myth does not require the deployment of special landmark nodes. Our trick is to make use of the existing Myth nodes, which are already in the system, as landmark nodes. This solution is scalable and free from single point of failure problem as in [2], since there are no fixed landmarks.

C. Myth Implementation

Algorithm 2 Myth Algorithm

```

Connect_to_Rendezvous_Point(rp)
Get_Neighbor_Candidates_List(rp)
Join_Overlay()
Make_Connections_to_Neighbors()
for  $i$  in  $Neighbors$  do
     $d(i)$  = Measure Distance to  $i$ 
end for
 $x_i$  =  $Initial\_Coordinates\_Prediction(d(\cdot), NCs(\cdot))$ 
Wait(Update_Interval)
while forever do
     $j$  = random(neighbors of  $i$ )
     $x_i$  =  $vivaldi(rtt, x_j, e_j)$ 
    Wait(Update_Interval);
end while

```

Algorithm 2 shows the procedure that a new node A joins a Myth overlay. Node A first contacts the Rendezvous Point (RP) of the Myth system like all the other P2P schemes. After obtaining a list of neighbor candidates from RP, Host A joins the overlay and measures its RTTs to L nodes in its neighbor set. Then A employs IC prediction algorithm to calculate its initial coordinates. After that node A can start the coordinates adjusting procedure and update its coordinates periodically using basic Vivaldi algorithm.

D. Overhead Analysis

By adding the IC prediction scheme, Myth introduces extra communicational and computational overhead when a new node joins. The computational overhead is negligible because the time needed to compute an IC is far shorter than the interval between update rounds, which is generally on the order of seconds. In this section we present a simple analysis of the communicational overhead imposed by Myth.

Suppose there are M nodes in the swarm, and each node probes one of its neighbors and updates its NC once per unit

T (units)	120	240	360
Time (minutes)	10	20	30
$(L-1)/T$	0.13	0.07	0.04

TABLE I
COMMUNICATIONAL OVERHEAD

time. Thus for each node the communicational load (number of messages per unit time) of Vivaldi is 1. We do not take into account the sizes of messages because almost same format of messages is used for all communications. During IC prediction, each node will contact extra $L - 1$ neighbors ($L < M$) to compute its IC (one neighbor information is shared with the first round of Vivaldi update). If one node's average staying duration is T , the upper bound of the extra communicational load introduced by IC prediction is $\frac{L-1}{T}$, where the IC prediction happens for each node once per T time units. In practical settings, L is 16, which is also the number of neighbors in our experiment, and we test multiple T 's to show different overheads under different node churn situations. The unit time is set to 5 seconds according to [3]. Table.I shows the results.

Even when the average node stay duration is as short as 10 minutes, the overhead of Myth is only increased by 13% of Vivaldi. If the average node stay duration reaches 30 minutes, Myth will adds merely 4% communicational overhead to Vivaldi. Thus the extra overhead is acceptable.

IV. PERFORMANCE EVALUATION

A. Experiment Setup

In our experiments, we use two data sets, the King and PlanetLab, to compare Myth and Vivaldi. Both systems employ 7-dimension coordinates, and each node has 16 neighbors in the overlay. c_c and c_e in Vivaldi and Myth are set to 0.25 as an empirical value in [3]. Our experiment is performed both in Stable scenario and Churn scenario. Ten runs are performed on each data set and the average results are reported.

B. Evaluation results of Myth

Relative Error: Fig.3 shows the comparison of relative error between Myth and Vivaldi. In Stable scenario, the performance of Myth is almost equivalent to that of Vivaldi. Hardly any difference can be found between the two curves. In Churn scenario, Myth outperforms Vivaldi a lot on both of the data sets. We pay more attention to the ninety-percentile relative error(NPRE) which would be helpful to NC-aware applications [2] [3]. On PlanetLab data set, the NPRE of Myth is 0.624 while Vivaldi's is 0.820. On King data set, NPRE of Myth is 0.718 and that of Vivaldi is 0.848. Myth can reduce the NPRE from Vivaldi by 23.90% in PlanetLab data set and 15.33% in King data set. Furthermore, the performance of Myth under Churn scenario is very close to the performances of Vivaldi and Myth under Stable scenario, which indicates that Myth can reduce the bad effect of the node churn, and maintain the accuracy in churn environments.

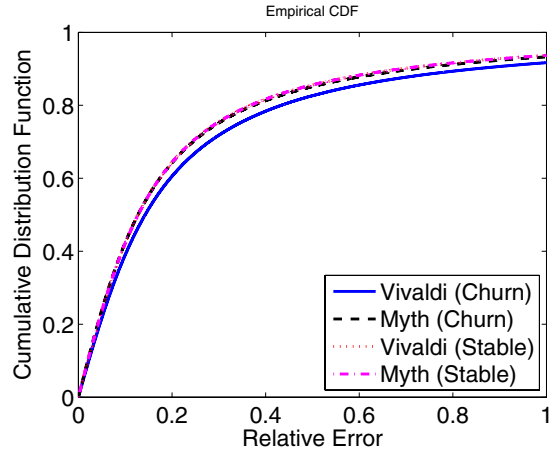
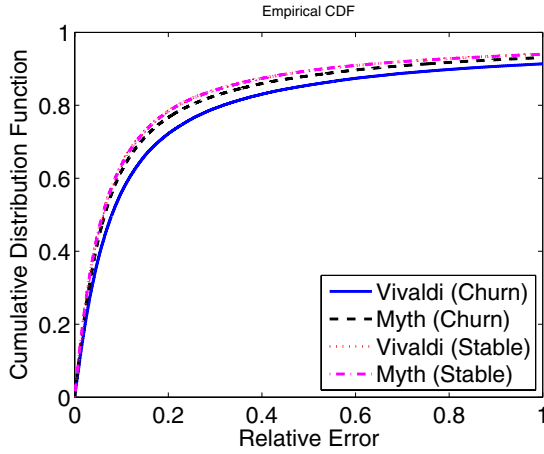


Fig. 3. Distribution of Relative Error

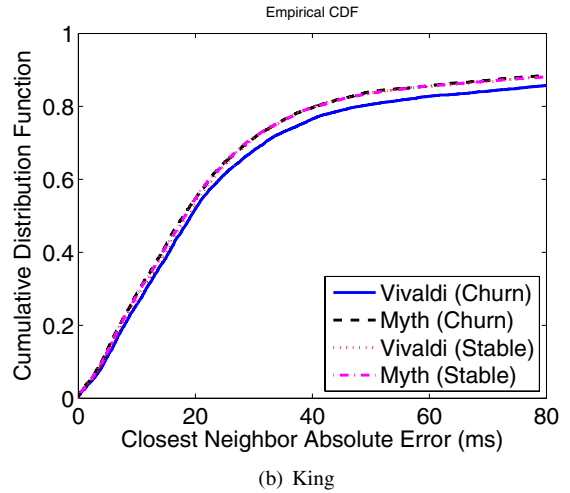
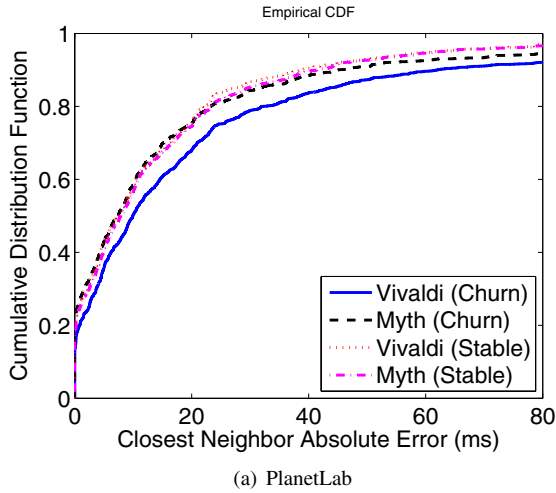


Fig. 4. Distribution of Closest Neighbor Absolute Error

Data set	Vivaldi (Stable)	Myth (Stable)	Vivaldi (Churn)	Myth (Churn)
PlanetLab	76.32%	77.21%	81.85%	74.69%
King	98.10%	98.31%	98.51%	97.93%

TABLE II
CLOSEST NEIGHBOR LOSS

C. Other Metrics

Besides RE described in Section II, we also evaluate the performance of Myth with the following three metrics.

Relative Rank Loss (RRL) [10] measures the probability to correctly select the closer node from an arbitrary node pair. It is defined as the percentage of incorrectly ordered node pairs (as perceived at a given node) based on the prediction.

Closest Neighbor Loss (CNL) [10] indicates the probability to correctly select the closest neighbor to a given node, and is defined by the fraction of nodes where an incorrect node is chosen as the closest neighbor using predicted distances.

In addition to the CNL, we also measure the magnitude of the error when the wrong node is selected. More precisely, we use *Closest Neighbor Absolute Error (CNAE)* [12], which is defined as the gap between the distance to the incorrectly selected neighbor and the distance to the actual closest neighbor.

Among these metrics, relative error (RE) is the basic metric which is evaluated by all NC designers. RRL, CNL and CNAE focus more on application perspective where nodes only need to know the relative distances of other nodes.

Closest Neighbor Loss and Closest Neighbor Absolute Error: Applications will benefit from the higher prediction quality on various aspects. For example, the lower CNL and CNAE are, the higher probability applications have to find the nearest neighbors. Table.II and Fig.4 show the comparison of CNL and CNAE between Myth and Vivaldi. In Stable Scenario, the CNAEs of Myth and Vivaldi are also very close to each other. In Churn Scenario, in both PlanetLab data set and King data

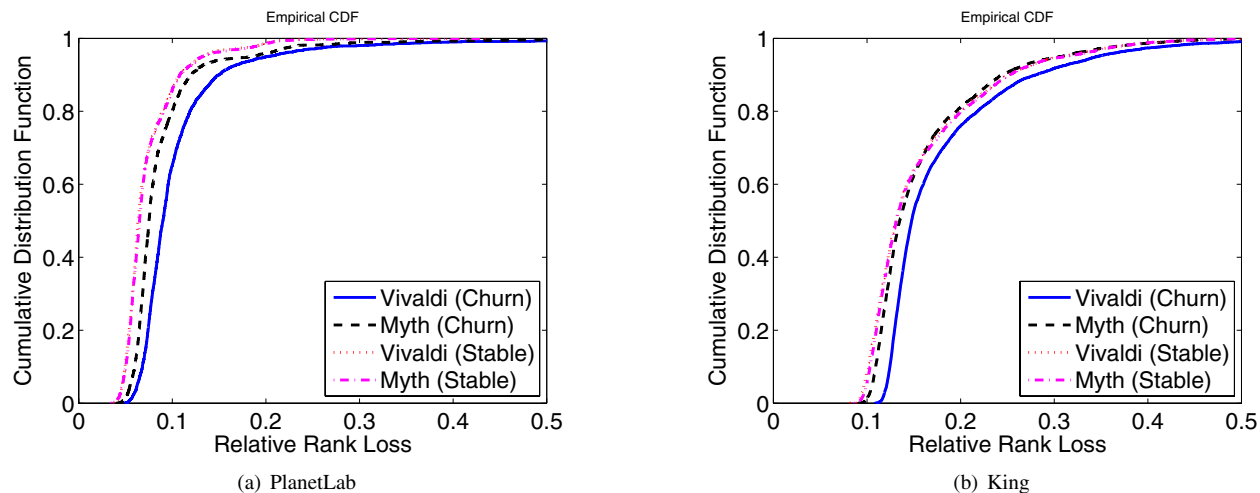


Fig. 5. Distribution of Relative Rank Loss

set, Myth improves the quality of the closest neighbor selection a lot comparing to Vivaldi. In addition, the performance of Myth under Churn scenario is very close to the performances of Vivaldi and Myth under Stable scenario.

Relative Rank Loss: Fig.5 shows the comparison of relative rank loss between Myth and Vivaldi. Just as the results in RE and CNL metrics, the RRL of Myth and Vivaldi in Stable Scenario are almost equivalent with both PlanetLab data set and King data set. Myth outperforms Vivaldi with both PlanetLab data set and King data set in Churn scenario. Also, the performance of Myth under Churn scenario is very close to the performances of Vivaldi and Myth under Stable scenario.

V. CONCLUSION

In this paper we firstly study the impact of high node churn rate to a representative distributed NC system, Vivaldi, and then propose an accurate and scalable NC system called Myth, to improve the prediction accuracy under high node churn rate. The major contribution of this paper is twofold. (1) By evaluating the performance of Vivaldi under high node churn rate, we find out that the NCs of Vivaldi nodes suffer from degradation of prediction accuracy under such situation. (2) We propose Myth, a fully decentralized NC scheme, to improve the accuracy of Internet distance prediction under high node churn rate. Myth is both accurate and scalable under high node churn rate, combining merits of LBA and SBA together without introducing any significant overhead. We evaluate the performance of Myth and compare it with the Vivaldi system with real Internet measurement traces. The experimental results show that Myth can remedy the node churn problem of Vivaldi. In our experiment, the performance of Myth exceeds that of Vivaldi a lot in Churn Scenario, and is almost equivalent to the performance of Vivaldi in the scenario where all nodes stay stable. In other words, Myth is robust to node churn.

To further evaluate the practicality of Myth, we currently focus on deploy Myth on Internet and develop some real applications based on this NC system.

ACKNOWLEDGEMENT

This work is supported by the National Science Foundation of China (No.60473087) and National Basic Research Program of China (No.2007CB310806). Thanks to Prof. Pei Cao from Stanford University for her comments and suggestions.

REFERENCES

- [1] P. B. Godfrey, S. Shenker, and I. Stoica. Minimizing Churn in Distributed Systems. In Proc. of ACM SIGCOMM, Sep 2006.
- [2] T. S. E. Ng and H. Zhang. Predicting Internet Network Distance with Coordinates-Based Approaches. In Proc. of IEEE INFOCOM, June 2002.
- [3] F. Dabek, R. Cox, F. Kaashoek, et al. Vivaldi: A Decentralized Network Coordinate System. In Proc. of ACM SIGCOMM, August 2004.
- [4] J. Ledlie, P. Pietzuch, and M. Seltzer. Stable and Accurate Network Coordinates. In Proc. of IEEE ICDCS 2006, July 2006.
- [5] J. Ledlie, P. Gardner, and M. Seltzer. Network Coordinates in the Wild. In Proc. of USENIX NSDI'07, Apr. 2007.
- [6] F. Wang, Y. Q. Xiong, J. C. Liu. mTreebone: A Hybrid Tree/Mesh Overlay for Application-Layer Live Video Multicast. In Proc. of IEEE ICDCS, Jun 2007.
- [7] M. Pias, J. Crowcroft, S. Wilbur, et al. Lighthouses for Scalable Distributed Location. In Proc. of IPTPS, February 2003.
- [8] Y. Mao, L. Saul, J. M. Smith. IDES: An Internet Distance Estimation Service for Large Networks. IEEE Journal On Selected Areas in Communications (JSAC), 24(12): pp 2273-2284, Dec 2006.
- [9] K. P. Gummadi, S. Saroiu, and S. D. Gribble. King: Estimating Latency between Arbitrary Internet End Hosts. In Proc. of ACM SIGCOMM IMW, November 2002.
- [10] E. K. Lua, T. Griffin, M. Pias, et al. On the Accuracy of Embeddings for Internet Coordinate Systems. In Proc. of ACM IMC, October 2005.
- [11] Azureus BitTorrent. <http://azureus.sourceforge.net/>.
- [12] R. Zhang, Y. C. Hu, X. Lin, and S. Fahmy. A Hierarchical Approach to Internet Distance Prediction. Technical Report TR ECE 06-03, Purdue University, March 2006.
- [13] NC Research Group at Harvard. <http://www.eecs.harvard.edu/~syrah/nc/>.
- [14] Y. Shavitt and T. Tankel. Big-Bang Simulation for Embedding Network Distances in Euclidean Space. In Proc. of IEEE INFOCOM, April 2003.
- [15] Y. H. Chu, A. Ganjam, T. S. E. Ng, et al. Early deployment experience with an overlay based Internet broadcasting system, in USENIX Annual Technical Conference, Jun. 2004.