# A Retrospective Approach for Accurate Network Latency Prediction

Fan Ping, Christopher McConnell, Jeong-Hyon Hwang
Department of Computer Science
University at Albany - State University of New York
Albany, NY 12222
Email: {apping, ctm, jhh}@cs.albany.edu

***Abstract*—Network coordinate systems can efficiently predict the round-trip time between any pair of nodes on the Internet. These systems map nodes onto a multi-dimensional coordinate space such that the round-trip time between two arbitrary nodes can be estimated by the distance between the corresponding pair of points in the coordinate space. In these systems, each node continuously updates its coordinates based on the latencies of new communications and the coordinates of other nodes. Node coordinates obtained in this manner, however, may be misled by erroneous information, such as extraordinary round-trip times. Previous techniques attempt to overcome this problem by having nodes reject new information if it is inconsistent with the current coordinates. These techniques are still inappropriate if the coordinates are not sufficiently accurate.**

**In this paper, we develop a new approach where each node preserves recent communication history. To update a node's coordinates more accurately, our approach re-evaluates the records in the node's history while ignoring inconsistent communication records and applying a higher weight to more reliable records. This approach thus can prevent node coordinates from being affected by erroneous information that was difficult to detect in the past. Through evaluations based on real network trace data, we demonstrate that our approach can reduce network latency prediction errors by 50% compared to previous approaches.**

## I. Introduction

Recently, there has been significant interest in techniques that can allow nodes on the Internet to determine the latency to any other node without actively monitoring all of the nodes. A variety of applications can benefit from these techniques. For example, a central system with clients in diverse geographic areas would service all the clients with low communication latency if the service is provided at the centroid of the clients. Furthermore, a client in a peer-to-peer system can efficiently locate a peer, among many peers that store the desired file, within the shortest communication delay. Other applicable areas include routing [1], content distribution [2], and distributed data processing [3].

Network coordinate systems aim to efficiently provide the aforementioned functionality [4], [5], [6], [7], [8], [9]. These techniques map nodes onto a multi-dimensional coordinate space such that the round-trip time between any pair of nodes can be estimated by the distance between the corresponding pair of points in the coordinate space. Due to this abstraction, network coordinate systems can allow us to solve various problems in distributed systems as geometric problems [6].

In previous network coordinate systems, adaptation to changing system conditions is done by consistently updating node coordinates based on the latencies of new communications. These techniques, however, may be misled by erroneous information, such as extra-ordinary round-trip times. One proposed technique to solve this problem requires each node to remember the round-trip times and to update its coordinates based on the 25th percentile value (for each connection to a remote node) [6]. Other techniques detect outliers based on geometric constraints, such as triangle inequality [10], [11]. Nevertheless, all of these methods have limitations. The former cannot effectively detect outliers if nodes do not communicate often with the same set of nodes. The latter may mistakenly regard valuable information as outliers if the current local coordinates have been chosen using inappropriate information.

In this paper, we propose a new technique, called Retrospective Network Positioning (RNP) that can tackle the above limitations. In RNP, each node preserves records about recent inter-node communications as well as their round-trip times. RNP can thus retrospectively update node coordinates while filtering out past information that was later found to be inconsistent. RNP is also designed based on the observation that the coordinates of a node may change significantly over time depending on the network condition as well as the load of the node. In RNP, stable node coordinates have a higher impact in coordinate calculation, thereby effectively bounding the latency prediction errors.

Through evaluation using real network traffic data, we demonstrate that RNP, compared to previous techniques, can achieve comparable prediction accuracy with significantly fewer inter-node communications. Further, we show that RNP guarantees more stable node coordinates than other techniques and thus can reduce the network latency prediction error by 50% in certain situations.

The remainder of this paper is organized as follows. In Section II, we provide an overview of previous techniques and discuss their limitations. In Section III, we present our RNP technique while highlighting its benefits. In Section IV, we substantiate the utility of RNP by contrasting it with previous techniques. Then, we summarize related work in Section V and conclude in Section VI.

## II. BACKGROUND

Network coordinate systems embed nodes onto a virtual coordinate space in a manner that facilitates network latency prediction. In this section, we provide overviews of a representative algorithm, called Vivaldi, that determines node coordinates (Section II-A) as well as methods that detect outliers (Section II-B). We also discuss the limitations of these previous techniques (Section II-C).

### A. Vivaldi

Vivaldi is a distributed algorithm that assigns synthetic coordinates to nodes [4]. In Vivaldi, nodes piggyback information about their coordinates onto messages that upper-level applications exchange. Whenever a node receives a message, it updates its coordinates in a manner that reduces the difference between the actual round-trip time and the distance to the remote node in the coordinate space.

**Algorithm 1** Vivaldi($rtt, x', e'$)

1: $x \leftarrow x + c_c \times \frac{e}{e+e'} \times (rtt - ||x - x'||) \times u(x - x')$
2: $\alpha \leftarrow c_e \times \frac{e}{e+e'}$
3: $e \leftarrow \frac{\left| ||x-x'|| - rtt \right|}{rtt} \times \alpha + e \times (1 - \alpha)$

Algorithm 1 shows how a node updates its coordinates based on the round-trip time ($rtt$), the remote node's coordinates ($x'$) as well as the error ($e'$) that remote node has derived from its past coordinate computation. In the algorithm, the node first computes the force vector $(rtt - ||x - x'||) \times u(x - x')$ that can rectify the mismatch between the round-trip time $rtt$ and the estimated round trip time $||x - x'||$ (line 1). Here, $u(x - x')$ denotes a unit-length vector that represents the direction of the force vector, while $\frac{e}{e+e'}$ represents the confidence of the local node regarding its coordinates relative to that of the remote node. Multiplying the force vector by $\frac{e}{e+e'}$ allows more confident nodes to tug harder than less confident nodes. Constant $c_c$ dampens the magnitude of the coordinate change to prevent the coordinates from oscillating.

In the algorithm, the node also updates its coordinate error $e$ using an exponential moving average of the relative prediction error $\frac{\left| ||x-x'|| - rtt \right|}{rtt}$ with weight $\alpha$ (line 3). $\alpha$ is in turn the product of a dampening constant $c_e$ and the confidence $\frac{e}{e+e'}$ of the node relative to the remote node (line 2).

### B. Outlier Filtering Methods

The Vivaldi algorithm described in Section II-A has been adopted in various contexts [12], [6], [13], [11], [14], [7]. A main limitation of Vivaldi is that node coordinates can be easily affected by abnormal round-trip times resulting from network congestion, server overload, and possibly attacks by malicious nodes that intentionally hinder message delivery. As a solution to this problem, Ledlie et al., have developed Moving Percentile (MP) filtering [6]. In this technique, nodes store the latencies of communications, for each remote node and then use a predefined percentile of the sorted latencies. This prevents outliers from being used in coordinate calculation.

Other classes of outlier filtering techniques that rely on node coordinates have also been developed. One such technique, Triangle Inequality Violation (TIV) Alert, classifies information about a communication as an outlier if the round-trip time is larger than a certain bound obtained from past communications [10], [11]. Wang et al.'s recent work [11] argues that TIV Alert outperforms other statistical methods [9], [15], [16]. We thus use TIV Alert as the representative of the coordinate-based techniques throughout this paper.

### C. Limitations of Previous Methods

As described in Section II-B, the prediction performance of Vivaldi degrades as the number of outliers increases. Techniques summarized in Section II-B also have limited outlier detection capability. In particular, MP filtering can detect outliers for each remote node only when multiple communications with that node have taken place. This technique thus cannot effectively detect outliers if nodes do not communicate often with the same set of nodes. Coordinate-based outlier detectors, represented by TIV Alert, may also mistakenly regard valuable information as an outlier if the current coordinates have been chosen using inappropriate information.

## III. RETROSPECTIVE NETWORK POSITIONING

Our retrospective network positioning technique, abbreviated as RNP, preserves records about recent inter-node communications. By re-examinating the validity of past communication records, RNP can accurately predict network latencies. In this section, we describe the operation of RNP (Section III-A) and highlight the advantages of RNP over previous network positioning techniques (Section III-B).

### A. The RNP Algorithm

The design of RNP is motivated by the limitations of previous network positioning techniques (Section II-C). In particular, we note that it is hard for a node to be confident about its position in the coordinate space if the node has rarely communicated with other nodes. For example, if a node has conducted only the first round of communication with a remote node, it cannot determine if the communication had an unusual latency. Further, if a node has not yet found an appropriate location in the coordinate space, it is not effective to detect outliers by comparing the round-trip time and the distance between the corresponding points of the nodes in the coordinate space. Despite such uncertainty, nodes must process incoming coordinate information. Otherwise, they cannot update their coordinates and thus cannot more accurately predict network latencies. This requirement, however, implies that node coordinates (particularly at an early) are susceptible to erroneous information, which can often be discovered only when more network measurement data becomes available. For this reason, each node in RNP preserves recent inter-node communications and re-evaluates them with forthcoming information.

Algorithm 2 describes the operation of RNP when a node receives a message from a remote node. The message contains

**Algorithm 2** RNP($id', rtt, x', e', v'$)

```
1:  H.insert[id', rtt, x', e', v', current_time()];
2:  if H.size() > max_H then
3:      H.remove_first();
4:  end if
5:  for r ∈ H do
6:      // r appears in the order of increasing (1/r.v', r.time)
7:      if !outlier(r) then
8:          Vivaldi(r.rtt, r.x', r.e');
9:      end if
10: end for
11:
12: outlier(r)
13: count ← 0;
14: for r' ∈ H do
15:     δ ← 1 + β(r.v' + r'.v');
16:     if (r.rtt + r'.rtt)δ < ||r'.x − r'.x'|| or
        |r.rtt − r'.rtt| > ||r'.x − r'.x'||δ then
17:         count ← count + 1;
18:     end if
19: end for
20: if count > θ_outlier then
21:     return true;
22: else
23:     return false;
24: end if
```

| Symbol | Meaning |
|---|---|
| $rtt$ | round-trip time of the last communication |
| $x, x'$ | coordinates of local and remote nodes, respectively |
| $e, e'$ | coordinate errors of local and remote nodes |
| $v, v'$ | recent coordinate velocities of local and remote nodes |
| $id'$ | identifier of the remote node |
| $\mathcal{H}$ | local node's comm. history: sorted by $(\frac{1}{v'}, time)$ |
| $max_{\mathcal{H}}$ | maximum size of $\mathcal{H}$ |
| $w$ | window size (# msgs) for velocity calculation |
| $\beta$ | fudge factor for triangular inequality violations |
| $\theta_{outlier}$ | threshold: if the number of TIVs is larger than this threshold, the given communication record is regarded as an outlier. |

TABLE I
SYMBOLS AND THEIR MEANINGS

the remote node's coordinates ($x'$) and error ($e'$) as in the original Vivaldi algorithm (Section II-A). It also includes the remote node's identifier ($id'$) as well as the recent velocity ($v'$) in the coordinate space. This velocity is computed by the remote node over the the most recent window of $w$ communications as follows:

$$v' := \frac{\sum_{i=1}^{\log_2 w} \frac{||\ x'[c]-x'[c-i]\ ||}{2^i}}{\log_2 w} \tag{1}$$

where $x'[c]$ and $x'[c-i]$ denote the remote node's current coordinates and those before $i$ communications, respectively. If the remote node has received fewer than $w$ messages, $v'$ is set to a sentinel value to indicate the lack of sufficient information. It should be noticed that $\frac{1}{v'}$ can used to indicate the *stability* of the remote node's coordinates. Table I summarizes all the symbols used in Algorithm 2.

The first step that the receiver node takes is to record the received information as well as the current time in the history buffer $\mathcal{H}$ (line 1). If the node has received multiple messages from a remote node (distinguished by $id'$), it uses the 25th percentile of the round-trip times to ignore outliers as in MP filtering [6]. Our implementation of $\mathcal{H}$ is based on a *sorted* tree structure that achieves $O(n)$ time cost for scanning the stored records and $O(\log n)$ time cost for update operations such as insert() and remove_first(), where $n$ is the number of records stored in $\mathcal{H}$.

Next, the node bounds the growth of history $\mathcal{H}$ based on a parameter $max_{\mathcal{H}}$, the maximum size of $\mathcal{H}$ (lines 2-4). When a record needs to be removed from $\mathcal{H}$, the record with the highest velocity (i.e., the record about least stable coordinates) is chosen. After this, the node adjusts its coordinates by re-processing the stored communication records (using Vivaldi) in a lexicographical order $(\frac{1}{v'}, time)$ (lines 5-10). The motivation for this ordering is that node coordinates have significantly different stability levels (e.g., overloaded nodes tend to have significantly varying coordinates) and processing more stable coordinates later will make these coordinates have a higher impact on the resulting coordinates. The latter is a consequence of using the exponential moving average model in Vivaldi (line 1 in Algorithm 1), which gives a higher weight to the records processed later. The second $time$ component in the ordering makes more recent coordinate information have a higher impact on the resulting coordinates.

RNP also includes an effective outlier filtering method (line 4; lines 12-24). Given a communication record $r$, this outlier filter checks how many records in $\mathcal{H}$ violate triangular inequalities when considered with $r$ (lines 13-19). If the number of violations (i.e., $count$) is larger than a pre-defined threshold $\theta_{outlier}$, then record $r$ is classified as an outlier (lines 20-24). In our current implementation, $\theta_{outlier}$ is set to one fifth of the number of records in $\mathcal{H}$. For the given record $r$ and each record $r'$ contained in $\mathcal{H}$, our outlier filter examines generalizations of two triangular inequalities (lines 16 and 17). $\delta$ in the inequalities is a fudge factor to tolerate slight violations of triangular inequalities. $\delta$ has a larger value (and thus can tolerate more triangular inequality violations) as the constant $\beta$ has a higher value or as the records $r$ and $r'$ become less reliable (i.e., the corresponding coordinate velocities increase).

### B. Discussion

As described in Section III-A, RNP keeps track of inter-node communications so as to accurately predict network latencies. If it turns out that a communication record is inconsistent with other communication records obtained later, RNP thus can rectify the node coordinates by skipping the erroneous record while updating the node coordinates. On the other hand, even if a communication record has been mistakenly classified as an outlier, the record can still be incorporated into future coordinate computation. Furthermore, RNP prioritizes communication records so that node coordinates can be updated based on more reliable coordinate information. Section IV demonstrates that due to the above characteristics, RNP can provide higher prediction accuracy than other techniques. We also show that RNP can achieve comparable prediction accuracy with substantially fewer inter-node communications used by previous techniques.

## IV. Evaluation

In this section, we present the results of our evaluation study on network latency prediction techniques. We describe the settings (Section IV-A) and compare RNP with previous techniques in terms of network latency prediction accuracy (Sections IV-B and IV-C).

### *A. Settings*

To evaluate RNP as well as a number of other representative techniques, we have developed an event-based simulator. We present the evaluation results obtained by replaying an online network traffic archive [17] collected over 283 nodes on PlanetLab [18].

In our simulations we focused on a subset of the data, utilizing 245 of the 283 nodes, selecting those with a full set of latency data upon formatting for our simulator. All of the 245 nodes initially have their synthetic locations at the origin of the coordinate space. Whenever nodes receive a message, they update their node coordinates based on the chosen network positioning technique.

In our simulations, we tested Vivaldi and RNP. Vivaldi and RNP were also tested with several outlier filtering techniques including MP filtering, TIV-Alert and our own (labelled TIV). Unless otherwise stated, we use the dimension of 3 plus the height vector (as in [4]) for the coordinate space and use the duration of 50 simulation epochs, each of which corresponds to a one hour period in the network traffic data [17]. During an epoch, each node sends/receives approximately 50 messages. All of the results presented in this section were collected from 15 simulation runs.

### *B. Prediction Accuracy Comparison*

The first set of evaluations compare the performance of network positioning techniques. For this comparison, we first obtained (from the network trace archive) the median of latencies for each pair of nodes. We then computed the relative prediction error for each pair of nodes. This error can be expressed as $\frac{rtt-||x-x'||}{rtt}$ where $x$ and $x'$ denote the coordinates of the sender and receiver nodes, respectively and $rtt$ denotes the corresponding median latency value. The median of these relative errors is used as a main performance metric throughout our study.

Figure 1 shows how the median of relative errors changes over time. In this figure, Vivaldi achieves limited relative prediction accuracy between 40% and 80%. A main reason behind this high error rate is that Vivaldi processes coordinate information without assessing the fidelity of the sender node.

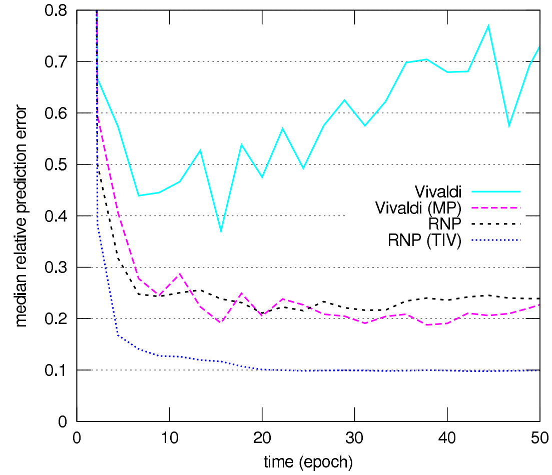


Fig. 1. Median Relative Prediction Error

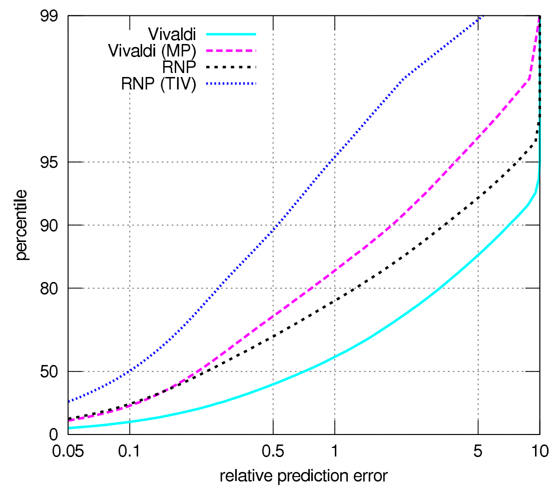


Fig. 2. Relative Prediction Error Distribution

Conversely, MP Filtering can reduce the relative prediction error down to 20-30% by filtering latency outliers. Our approach, RNP, can achieve similar results without using any outlier filters. Finally, we apply our TIV outlier detector to RNP reducing the relative prediction error to 10%, a 50% reduction compared to the best case of the Vivaldi implementation. It is important to note that a combination of TIV and Vivaldi provides no improvement in the prediction accuracy and thus has been omitted from the figure.

Figure 2 shows the distribution of relative prediction errors for all the node pairs. This figure shows that the median relative prediction error under RNP (TIV) is roughly 10%, which is achieved around 20 epochs (Figure 1). Furthermore, this figure shows that RNP (TIV) leads to moderate worst cases. For example, under RNP (TIV), about 90% of all the node pairs had relative prediction errors less than 50%. In contrast, under Vivaldi (MP) only 71% of the node pairs could achieve comparable prediction accuracy. The figure also shows that RNP (TIV) can substantially outperform other alternatives at the 90th percentile of relative prediction errors (RNP (TIV):

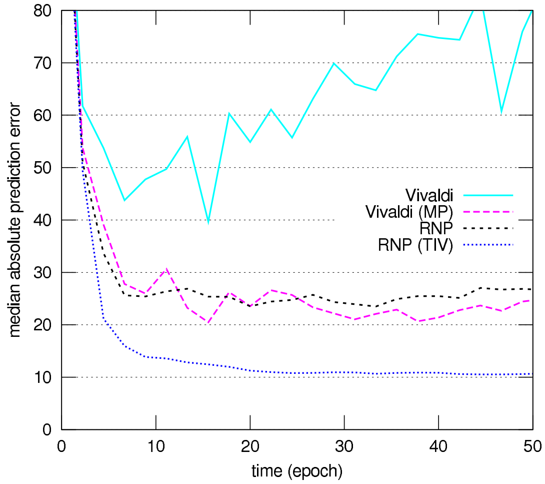


Fig. 3. Median Absolute Prediction Error

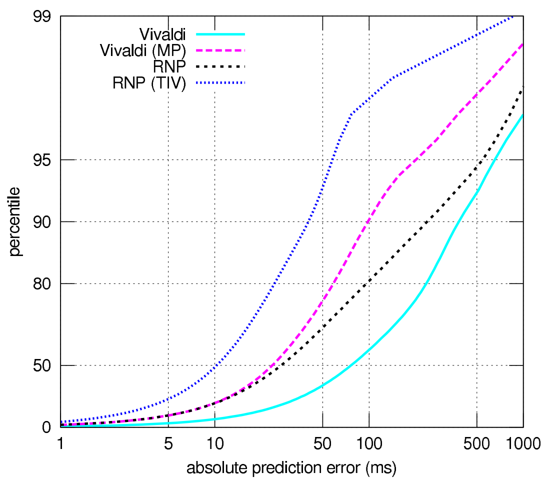


Fig. 4. Absolute Prediction Error Distribution

56%, Vivaldi (MP): 175%, Vivaldi: 880%). This implies that RNP (TIV) can keep prediction accuracy under control even in unfavorable situations.

Figures 3 and 4 show the variation of median absolute prediction error over time and the distribution of absolute prediction errors (in ms) for all the node pairs, respectively. As can be seen in both figures, the median absolute prediction error under RNP (TIV) is roughly 10ms, where as that of Vivaldi (MP) and Vivaldi are (roughly) 26ms and 95ms, respectively. Furthermore, it can been seen from Figure 4 that RNP (TIV) leads to acceptable worst cases. For example, RNP (TIV) can substantially outperform other alternatives at the 90th percentile of absolute prediction errors (RNP (TIV): 45ms, Vivaldi (MP): 97ms, Vivaldi: 432ms).

### *C. Comparison of Outlier Filters*

Figure 5 shows the impact of different outlier detection techniques on RNP. RNP performs well with all tested outlier detectors, as the worst case median perdiction error is approximatly 19%, due to the re-evaluation of historical data.

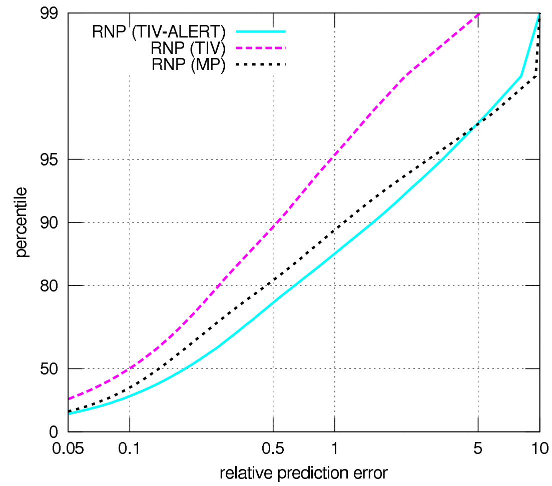


Fig. 5. Comparison of Outlier Filters

Our TIV outlier filter outperforms previous outlier detection techniques in the worst case scenario by a significant margin. For example, at the 90th percentile, RNP (TIV) has a relative prediction error of only 50%, however RNP (TIV-Alert) and RNP (MP) have relative prediction errors of 176% and 128% respectivly. This implies that our TIV outlier detection technique can better handle inconsistency in the communication records.

## V. Related Work

Recently, there has been a significant volume of research done on network latency prediction. In Section II, we summarized recent studies that have high relevance to our work. In this section, we provide a broader overview of related work.

**Fundamental Network Prediction Techniques:** IDMaps is one of the earliest systems that can predict network latencies between arbitrary nodes. It first obtains network latency data for all pairs of predetermined landmark nodes followed by each node obtaining the latency information about the closest landmark. IDMaps uses this combined data for latency prediction [5]. The King method operates in a way similar to IDMaps, using DNS servers as landmarks [19]. GNP extends IDMaps by embedding the landmarks, as well as nodes, into an Euclidean coordinate space [8]. GNP predicts the latency between any two nodes based on the distance between their corresponding points in the coordinate space. Vivaldi, which improves on GNP, is a fully decentralized network positioning that does not rely on landmarks [4]. Further details of Vivaldi were presented in Section II-A of this paper.

**Enhancement Techniques:** In recent years, many researchers have extended the aforementioned techniques. For example, Tang et al. adopted PCA to reduce the dimensionality of the GNP coordinate space [20]. The authors argued that a 7-9 dimension Euclidean space can provide sufficient prediction accuracy. PIC [21] and PCoord [22] are network positioning techniques which enable nodes to achieve good prediction accuracy by contacting a subset of predetermined landmarks.

As summarized in Section II-B, several outlier filtering techniques have been developed [6], [7], [10], [11]. These techniques commonly strive to prevent outliers from degrading the prediction accuracy. The inventors of the techniques also considered the coordinate drifting problem that has been often observed in real applications. Ledlie et al. use a gravity approach to offset the movement of the centroid the nodes [7], but this method seems impractical if the network conditions unpredictably change over time. Conversely, Wang et al. allow coordinate updates only if the change in coordinates is larger than a predefined threshold [11], however the authors showed that this method may not improve prediction accuracy. In contrast to these techniques, our RNP technique can enhance the prediction capability while continuously adapting to changing system conditions.

Kaafar et al. have taken into account security issues that might arise in network coordinate systems [23], [24], [15], [16], [25]. The authors used the Kalman filter to identify malicious nodes [16]. In contrast, Zage et al. used Mahalanobis distance function [9] while Saucez introduced an authentication technique for the same purpose [13]. We leave the problem of extending RNP to deal with malicious attacks as our future research.

There have been other proposals on improving and measuring prediction accuracy. Chen proposed Pharos, which creates two separate coordinate spaces, one for short range latency prediction and the other for long range prediction [14]. Lua et al. claimed that usual relative prediction errors cannot fully represent the performance of an embedding system and proposed relative rank loss (rrl) as a performance indicator [26].

## VI. Conclusions

In this paper, we proposed a new technique called RNP, that can both efficiently and accurately predict the round-trip time between any pair of nodes on the Internet. We have shown that RNP achieves higher prediction accuracy using less internode communication than previous techniques. To achieve this higher degree of accuracy, RNP retains messages at each node, which are used to filter out any inconsistent information.

We are currently exploring techniques that will allow nodes newly joining the system to accurately predict network latencies with only a small number of communications. We also plan to expand the evaluation of RNP by running our simulation with a larger set of test data and executing an actual system on PlanetLab. The current design of RNP does not have a strong emphasis on malicious attacks in which nodes disseminate false information. Developing sophisticated measures against such attacks is also left as our future work.

## Acknowledgements

This work is supported in part by the University at Albany through the Faculty Research Awards Program (FRAP) - Category A.

## References


[1] I. Abraham and D. Malkhi, "Compact routing on euclidian metrics," *in Proc. of PODC*, pp. 141–149, 2004.

[2] M. Szymaniak, D. L. Presotto, G. Pierre, and M. van Steen, "Practical large-scale latency estimation," *Computer Networks*, vol. 52, no. 7, pp. 1343–1364, 2008.

[3] P. Pietzuch, J. Ledlie, J. Shneidman, M. Roussopoulos, M. Welsh, and M. Seltzer, "Network-Aware Operator Placement for Stream-Processing Systems," *in Proc. of ICDE*, pp. 49–58, 2006.

[4] F. Dabek, R. Cox, F. Kaashoek, and R. Morris, "Vivaldi: A decentralized network coordinate system," *in Proc. of SIGCOMM*, pp. 15–26, 2004.

[5] P. Francis, S. Jamin, C. Jin, Y. Jin, V. Paxson, D. Raz, Y. Shavitt, and L. Zhang, "Idmaps: A global internet host distance estimation service," *in Proc. of INFOCOM*, pp. 210–217, 2000.

[6] J. Ledlie, P. R. Pietzuch, and M. I. Seltzer, "Stable and accurate network coordinates," *in Proc. of ICDCS*, p. 74, 2006.

[7] J. Ledlie, P. Gardner, and M. I. Seltzer, "Network coordinates in the wild," *in Proc. of NSDI*, 2007.

[8] T. S. E. Ng and H. Zhang, "Predicting internet network distance with coordinates-based approaches," *in Proc. of INFOCOM*, pp. 170–179, 2002.

[9] D. J. Zage and C. Nita-Rotaru, "On the accuracy of decentralized virtual coordinate systems in adversarial networks," *in Proc. of CCS*, pp. 214–224, 2007.

[10] G. Wang, B. Zhang, and T. S. E. Ng, "Towards network triangle inequality violation aware distributed systems," *in Proc. of IMC*, pp. 175–188, 2007.

[11] G. Wang and T. S. E. Ng, "Distributed algorithms for stable and secure network coordinates," *in Proc. of IMC*, pp. 131–144, 2008.

[12] http://azureus.sourceforge.net/.

[13] D. Saucez, B. Donnet, and O. Bonaventure, "A reputation-based approach for securing vivaldi embedding system," *in Proc. of EUNICE*, pp. 78–85, 2007.

[14] Y. Chen, "Pharos: Accurate and decentralized network coordinate system," *IET Communications*, vol. 3, pp. 539–548.

[15] M. A. Kaafar, L. Mathy, C. Barakat, K. Salamatian, T. Turletti, and W. Dabbous, "Securing internet coordinate embedding systems," *in Proc. of SIGCOMM*, pp. 61–72, 2007.

[16] M. A. Kâafar, L. Mathy, K. Salamatian, C. Barakat, T. Turletti, and W. Dabbous, "Securing internet coordinate systems," *in Proc. of AINTEC*, pp. 167–168, 2007.

[17] http://www.eecs.harvard.edu/~syrah/nc/.

[18] PlanetLab, http://www.planet-lab.org.

[19] K. P. Gummadi, S. Saroiu, and S. D. Gribble, "King: Estimating latency between arbitrary internet end hosts," *in Proc. of SIGCOMM Internet Measurement Workshop*, 2002.

[20] L. Tang and M. Crovella, "Virtual landmarks for the internet," *in Proc. of IMC*, pp. 143–152, 2003.

[21] M. Costa, M. Castro, A. Rowstron, and P. Key, "Pic: Practical internet coordinates for distance estimation," *in Proc. of ICDCS*, pp. 178–187, 2004.

[22] L. wei Lehman and S. Lerman, "Pcoord: Network position estimation using peer-to-peer measurements," *in Proc. of NCA*, pp. 15–24, 2004.

[23] M. A. Kaafar, I. S. Antipolis, and L. Mathy, "Real attacks on virtual networks: Vivaldi out of tune," *in Proc. of LSAD*, pp. 139–146, 2006.

[24] M. A. Kâafar, L. Mathy, T. Turletti, and W. Dabbous, "Virtual networks under attack: disrupting internet coordinate systems," *in Proc. of CoNEXT*, p. 12, 2006.

[25] M. A. Kâafar, B. Gueye, F. Cantin, G. Leduc, and L. Mathy, "Towards a two-tier internet coordinate system to mitigate the impact of triangle inequality violations," *in Proc. of Neworking*, pp. 397–408, 2008.

[26] E. K. Lua, T. G. Griffin, M. Pias, H. Zheng, and J. Crowcroft, "On the accuracy of embeddings for internet coordinate systems," *in Proc. of IMC*, pp. 125–138, 2005.