Published in IET Communications
Received on 3rd April 2008
Revised on 19th August 2008
doi: 10.1049/iet-com.2008.0187

ISSN 1751-8628

Pharos: accurate and decentralised network coordinate system

Y. Chen[1] *Y. Xiong*[2] *X. Shi*[1] *J. Zhu*[3] *B. Deng*[1] *X. Li*[1]

[1]*Department of Electronic Engineering, Tsinghua University, Beijing 100084, People's Republic of China*
[2]*Wireless and Networking Group, Microsoft Research Asia, Beijing 100080, People's Republic of China*
[3]*Department of Engineering Science, University of Oxford, Parks Road, Oxford OX1 3PJ, UK*
E-mail: chenyang04@mails.tsinghua.edu.cn

Abstract: Network coordinates (NC) system is an efficient mechanism for Internet distance prediction with scalable measurements. The intrinsical cause for the unsatisfactory accuracy of the simulation-based NC algorithms has been identified. Then Pharos, a fully decentralised and hierarchical scheme, is proposed to solve this problem. Pharos leverages multiple coordinate sets at different distance scales, with the right scale being chosen for prediction each time. We evaluate the performance of Pharos system with the King data set and latency data from PlanetLab, and compare it with the representative NC system, Vivaldi. The experimental results show that Pharos greatly outperforms Vivaldi in Internet distance prediction without adding any significant overhead. Our extensive evaluation results also demonstrate that Pharos can significantly improve the performance in distributed Internet applications, such as overlay multicast and server selection.

1 Introduction

Optimisation of distributed applications requires knowledge of network link properties such as end-to-end latency. Latency can be measured directly via end-to-end probes, inferred indirectly (e.g. via DNS queries [1]), or estimated using network coordinates (NC) systems. NC system is an efficient mechanism to predict latency between any two Internet nodes without direct measurement. In the NC system, each node is assigned with a set of numbers called coordinates, and the network distance (latency) between any two nodes can be calculated from their coordinates with a distance function. Thus, NC system significantly reduces the active probing overhead and particularly benefits large-scale Internet applications. To date, network coordinates have demonstrated their merit in a wide variety of applications ranging from overlay multicast [2], server selection [3] distributed query optimisation [4], file-sharing via BitTorrent [5] and compact routing [6].

Several approaches in designing an NC system have been proposed in the literature, and they can be categorised into two classes [7, 8], namely landmark-based algorithm (referred as LBA in this paper) and simulation-based algorithm (referred as SBA). In LBA systems [9–11], each node measures its distance from a set of reference nodes called landmarks, and then its coordinates can be determined by minimising the difference between the actual distance (by measurement) and calculated distance from these landmarks. As for SBA systems, such as [12, 13], nodes and pairwise distances are mapped into a physical system, so nodes' coordinates can be determined by minimising the energy state of the whole simulated physical system.

Recent studies have revealed that both LBA and SBA systems are not satisfactory in terms of accuracy, which is often denoted by the relative error (RE) of prediction. RE is calculated pairwisely over the nodes set across all distance ranges, while in [14] the authors found out that in an LBA system, landmarks' distribution could greatly affect the prediction accuracy, and different landmark sets should be chosen for different ranges of distance to improve the overall accuracy.

However, in SBA, there is no global landmark in the system; the impact of the range of distance for the prediction accuracy

remains unknown. Thus, in this paper, we want to answer the following questions.

1. Does the range of distance of different peers affect the accuracy of prediction in SBA and how?

2. Without the hierarchy of landmarks as in LBA, how to design a distributed and efficient architecture for peers in SBA to remedy this problem?

The answer to the first question is positive, which has been confirmed with a series of experiments. We analyse the distribution of the relative error of a representative SBA system, Vivaldi. In contrast to the observation for the LBA systems [14], we have found out that the narrower range of distance does not contribute to the higher prediction accuracy in Vivaldi, whereas short links still suffer from higher prediction error than long links.

As for the second problem, our main idea is to cluster the peers into different groups with different ranges of distance, and design two sets of coordinates, which are more accurate for short and long distance, respectively. Then, according to the range of distance of interest, we can choose the right set of coordinates to achieve higher prediction accuracy. Note that the number of coordinate sets can vary depending on the scale of the range of distance. Based on this idea, we propose Pharos, a decentralised and hierarchical network coordinate system, and present its design and implementation. The distance prediction accuracy of Pharos system is evaluated with different typical Internet data sets. Our experimental results demonstrate the significant improvement of the prediction accuracy when compared with the original Vivaldi system. Furthermore, we study the performance of Pharos in two representative NC-based Internet applications: overlay multicast and server selection. The experimental results also indicate that the performance of these applications can be largely improved by using Pharos instead of Vivaldi.

The rest of this paper is organised as follows. In Section 2, some related works are reviewed. Then, the impact of range of distance and distribution on prediction accuracy is studied in Section 3 for a representative SBA system, Vivaldi. In Section 4, we present the design and implementation of Pharos, followed by its performance evaluation in Section 5. In Sections 6 and 7, we evaluate Pharos from the application perspective and study its performance in two representative NC-based Internet applications. Section 8 is the conclusion.

2 Background

2.1 Definition of network coordinates

Suppose we have N Internet nodes. Let S be the set of these N nodes. Let L be the $N \times N$ distance matrix among the nodes in S. Thus, $L(i, j)$ represents the measured round-trip time (RTT) between node i and node j.

Basically, NC is an embedding of these N hosts into m-dimensional Euclidean space R^m. We define x_i as the NC of node i, we have $x_i = (r_1^i, r_2^i, \ldots, r_m^i)$, $r_k^i \in R$, $1 \le k \le m$.

We can use the x_i and x_j to predict the RTT between node i and node j. We use $L^E(i, j)$ to represent this predicted RTT. The definition of $L^E(i, j)$ is as follows

$$L^E(i,j) = \|x_i - x_j\| = \sqrt{\sum_{1 \le k \le m} (r_k^i - r_k^j)^2} \quad (1)$$

To serve thousands of nodes effectively, an NC system should be scalable. Each node only does restricted measurements to calculate its NC. Thus, the system uses $O(N)$ measurements [9–13]. This total measurement overhead is much lower than the $O(N^2)$ measurements required for a full mesh of N nodes.

The prediction accuracy of a network coordinates scheme is often denoted by the relative error (RE) of predicted distance over the real latency measured on Internet. RE between node i and node j is defined as [10, 14–17]

$$RE = \frac{|\, L^E(i,j) - L(i,j) \,|}{L(i,j)} \quad (2)$$

Smaller RE indicates higher prediction accuracy. When measured latency is equal to predicted latency, the RE value will be zero.

2.2 Related network coordinate systems

Several algorithms for calculating network coordinates have been proposed. There are two classes of algorithms: landmark-based and simulation-based.

In landmark-based algorithms (LBAs), such as GNP [9], Virtual Landmark [17], ICS [18], IDES [11], a number of nodes called landmarks are introduced to serve as reference points for other nodes to calculate their coordinates. In GNP, nodes' coordinates are computed using the Simplex Downhill method. Lighthouse derives node coordinates by solving systems of linear equations. IDES exploits matrix factorisation to compute an incoming and an outgoing coordinate for each node. LBA provides high accuracy and stability, but it needs to deploy dedicated landmark nodes whose loads are rather heavy to serve all the nodes in a large-scale system. This will limit the scalability of the system.

Simulation-based algorithms (SBAs), such as Vivaldi [12] and Big Bang Simulation [13], determine coordinates using spring-relaxation and force-field simulation, respectively. In both systems, the nodes self-organise into overlay network, attracting and repelling each other according to network

doi: 10.1049/iet-com.2008.0187

distance measurements. The low-energy state of the physical system corresponds to the coordinates with minor error. SBA systems distribute the computation and measurement to all participating nodes, so the load of each node is rather light. However, in SBA, it takes many rounds for a node to update its own coordinates before converging to the ideal position, where energy of the whole system is the lowest. For Vivaldi, the convergence time of a Vivaldi node will take tens of seconds even when nodes stay stable in the system [19], thus the joining or leaving of each node will affect the whole system. If the Vivaldi nodes are under high churn rate (nodes join or leave frequently), the accuracy of NC will decrease [7, 19].

3 Impact of range of distance for Vivaldi

In our work, we focus on the improvement of accuracy of Vivaldi [12], which is known as the most widely used SBA system in [8] because of its clean and decentralised implementation. Vivaldi is studied in [20, 21] as the representative NC algorithms, and it is deployed in many well-known Internet systems, such as Bamboo DHT [22], Stream-based overlay network (SBON) [4] and Azureus BitTorrent [5]. Before going into the details, we first briefly introduce the basic procedure of Vivaldi, and then we study the causes of the prediction error of Vivaldi.

3.1 Vivaldi

Vivaldi characterises the whole network as a spring system. Let L_{ij} be the actual distance (RTT) between node i and node j in Vivaldi system, and x_i be the coordinate assigned to node i. The coordinates of a node are the result of minimising the following error function, which corresponds to the lowest energy

$$E = \sum_i \sum_j (L_{ij} - \| x_i - x_j \|)^2 \quad (3)$$

where $||x_i - x_j||$ is the distance between node i and node j in the chosen coordinates space.

In a decentralised Vivaldi version, each node owns coordinates x_i and local error e_i. All nodes adjust their network coordinates and local errors via measuring their latencies to some other nodes in the system. The pseudo-code for Vivaldi is shown as follows. c_e and c_c are tunable parameters.

Algorithm 1 *vivaldi*(rrt, x_j, e_j)
1: $w = e_i/(e_i + e_j)$
2: $e_s = |\|x_i - x_j\| - rrt|/rrt$
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: $x_i = x_i + \delta \times (rrt - \|x_i - x_j\|) \times u(x_i - x_j)$

A sample weight is first 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).

3.2 Impact of range of distance on Vivaldi

We use two different data sets from real Internet measurement to study the prediction error of Vivaldi. The first data set is the King data set from [12], which includes the round-trip latencies among 1740 Internet naming servers. The second data set, the PlanetLab data set, includes the round-trip latencies among 226 hosts on the PlanetLab [23]. This data set is downloaded from Network Coordinate Research Group at Harvard [24].

Fig. 1 depicts the link distance distribution for these two data sets; we can see that the distance range of the PlanetLab data set is wider than that of the King data set. However, in contrast to the observation for LBA systems where wider range of distance always results in higher prediction error, in our experimental result shown in Fig. 2, we find that the prediction accuracy with PlanetLab data set is higher than that with King data set.

We then further study how distance affects the prediction accuracy, with Fig. 3 depicting the distribution of the RE over the distance spectrum. We found that the RE varies from different ranges of distance. But, for both King data set and PlanetLab data set, it can be seen that short links suffer from much higher relative error than long links. Therefore we would like to design a system to improve the prediction accuracy of short links without sacrificing the prediction accuracy of long links.

Lua *et al.* [15] proposed a scheme to get better prediction for short links using the Vivaldi algorithm. We assume that

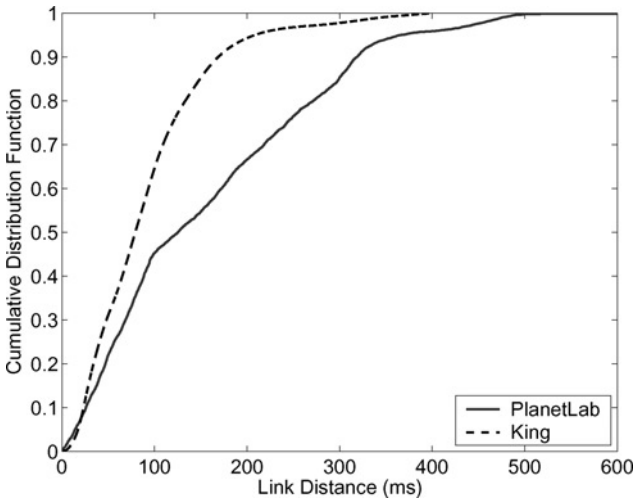

Figure 1 *Distribution of link distance*

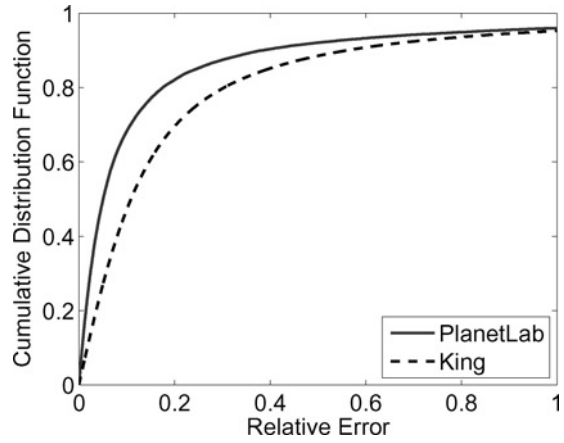

Figure 2 *Distribution of relative error*

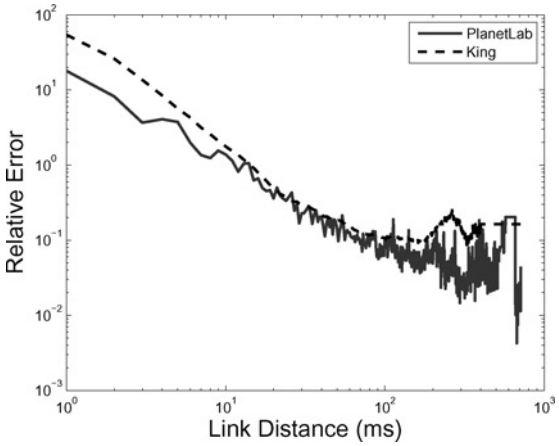

Figure 3 *Relationship between link distance and relative error*

we have a set of N nodes named S, then we have a subset of S named S_1, $S_1 \subset S$. If we want to predict the distance between two nodes both belonging to S_1, we can apply the Vivaldi algorithm to all N nodes of S, then use the global Vivaldi coordinate of these two nodes to calculate. In contrast, we can only apply the Vivaldi algorithm to all nodes of S_1, then use this local Vivaldi coordinate of these two nodes to calculate. The first way is named Superspace embedding and the second way is named Subspace embedding. According to their experiments, Subspace embedding is more accurate than Superspace embedding in Vivaldi. This result also provides us a good heuristic to use hierarchical Vivaldi to predict both the distances of short links and long links accurately. Detailed analysis is given in the following section.

4 System design

4.1 Pharos overview

In this section, we present a new approach, called Pharos, which exploits different sets of coordinates for the same node. Each node is assigned multiple coordinate sets, some of which positions the node at global scale, while the others position the node at a smaller distance scale. For simplicity, we use two sets of coordinates in Pharos, while the number of sets can be varied according to the scale of range of distance. All nodes in Pharos form two levels of overlays, namely base overlay for long link prediction and local cluster overlay for short link prediction. Two types of connections can then be defined accordingly: base overlay connections, which are constructed between nodes and their randomly selected neighbours in the Pharos overlay, and local cluster connections, which are constructed between nodes and their randomly selected neighbours in the same local cluster.

In Pharos, nodes must join both base overlay and local cluster to have two coordinates in different prediction scales. To join the base overlay, nodes can follow the procedure presented in [12] and create connections randomly to neighbours in the base overlay. To form the local cluster, we use a method similar to binning [3] and choose some nodes called anchors to help node clustering. This method only requires a one-time measurement (with possible periodic refreshes) by the client to a small, fixed set of anchors. Any stable nodes that are able to response ICMP ping message can serve as anchor, such as the existing DNS servers. For each anchor, we assign an identifier named AID to it. Guided by the anchors, nodes are grouped into different clusters as follows. A new node measures its distance to all the anchor nodes, finding out the nearest anchor, and joining the cluster led by it. Finally, the joining procedure of the corresponding cluster also follows [12] and the node will create connections randomly to neighbours in the same cluster as well.

Vivaldi algorithm is applied to both base overlay and local cluster. As a result, each Pharos node has two sets of coordinates. The coordinates calculated in the base overlay, which we call global NC, is used for the global scale, and the coordinates calculated in the corresponding local cluster, which we call local NC, covers a smaller range of distance.

4.2 Workflow of Pharos

Algorithm 2 shows the procedure that a new node A joins a Pharos overlay. Node A first contacts the rendezvous point (RP) of the Pharos system like all other P2P schemes. After obtaining a list of anchors from RP, Host A measures the distance to all the anchors and chooses the nearest cluster to join.

To announce its existence to the RP of the Pharos system, each node generates a membership message to the RP periodically. The membership message is a vector with four elements. The definition of this vector is as follows

$$Membership\ Message = < ID, IP, AID, TTL > \quad (4)$$

doi: 10.1049/iet-com.2008.0187

In the membership message vector, *ID* is the identifier of the node, *IP* is the IP address of the node, *AID* is the identifier of the anchor closest to the node, *TTL* records the remaining valid time of the message.

Then, node A joins both base NC overlay and local NC cluster through gossip [25] protocol. After that, node A can participate in the NC calculating procedure in both base overlay and local cluster, and update their coordinates periodically.

4.3 Hierarchical distance prediction

After getting the global NC and local NC, we can predict the distance between any two nodes. Distance prediction proceeds in a bottom-up fashion. If two nodes belong to the same local cluster, this implies that if they are relatively close to each other, the distance between them is predicted by local NC. Otherwise, if these two nodes belong to two different clusters, the distance between them is predicted by global NC. This hierarchical approach would promote the accuracy of the distance prediction. We say that the cluster node A belongs to C_A, the cluster node B belongs to C_B. The predicted distance of node A and node B is calculated as follows

$$D^E(A, B) = \begin{cases} \| x_{A.\text{local}} - x_{B.\text{local}} \| & C_A = C_B \\ \| x_{A.\text{global}} - x_{B.\text{global}} \| & C_A \neq C_B \end{cases} \quad (5)$$

Algorithm 2 Pharos

```
Connect_to_Rendezvous_Point(rp)
Get_Anchors_List(rp)
Nearest_Anchor_Distance = ∞
for i in Anchors do
   d(i) = Measure Distance to i
   if Nearest_Anchor_Distance > d(i) then
      Nearest_Anchor_Distance = d(i)
      Nearest_Anchor = i
   end if
end for
Join_Base_Overlay()
Join_Cluster(Nearest_Anchor)
while forever do
   j = random(local neighbors of i)
   x_i.local = vivaldi(rtt, x_j.local, e_j.local)
   Wait(Update_Interval);
   j = random(global neighbors of i)
   x_i.global = vivaldi(rtt, x_j.global, e_j.global)
   Wait(Update_Interval);
end while
```

Fig. 4 shows the predicted distance calculation policy of Pharos. For example, both Node A and Node B belong to the cluster led by anchor 1, as a result, the $D^E(A, B)$ is calculated by $\| x_{A.\text{local}} - x_{B.\text{local}} \|$. In contrast, node C belongs to the cluster led by anchor 2, node D belongs to the cluster led by anchor 3, they are not in the same cluster, thus the $D^E(C, D)$ is calculated by $\| x_{C.\text{global}} - x_{D.\text{global}} \|$.

4.4 Anchors: practical infrastructure for distributed node clustering

In [14], the authors studied the range of distance problem for landmark and explored constructing a landmark hierarchy that is shared by all nodes to improve the prediction accuracy. Specifically, a number of landmark nodes form a hierarchy through recursive clustering. Each cluster consists of landmark nodes that are close to each other.

A key difference between anchors in Pharos and landmarks in [14] is whether they are passive or active. In other words, landmarks in [14] should actively participate in the system, which means people must deploy NC client on landmarks for NC calculation. In contrast, the only requirement for anchors in Pharos is to reply ICMP PING request. Thus, we can choose the existing Internet servers as anchors in Pharos, because they only need to reply the PING query passively, and we do not need to deploy Pharos clients on these anchors. This makes Pharos much more practical.

In the implementation of Pharos, DNS servers are used as anchors for their stability. Before Pharos system starts, we run a selection procedure to obtain a list of the anchor candidates. The method is proposed in [26]. First, we randomly generate 100,000 IP address drawn from the prefixes announced in BGP as published by the Route Views project [27]. For each generated IP address k, a reverse DNS lookup is performed to get the associated DNS servers and a set of DNS address is returned named D_k. For each D_k, we simply perform ICMP PING request to each server that belongs to it and remove the servers that are not available to reply ICMP PING request. Afterwards, if the set is empty, it will be discarded. In addition, for a set with more than one server, this set will be kept if and only if all the servers in the set are topologically close to each other. As in [26], we perform traceroute to all the servers in this set. If different measurement samples are measuring the same network, we can confirm that these servers are physically co-located. Finally, for each set, we choose one server randomly to form the list of the anchor candidates. This procedure will keep the nodes that may provide inconsistent PING results out of the list.

Moreover, the hierarchical landmark approach in [14] needs a large number of landmarks for effectively improving prediction accuracy. The number of landmarks is exponential to the number of hierarchy level. Even for a two-level hierarchy, which is studied in [14], 256 landmarks are needed. Since landmark is a critical issue in LBA systems, the deployment of large number of landmarks would become a heavy burden for the NC system designers to maintain the reliability and load balance of so many nodes in the world.

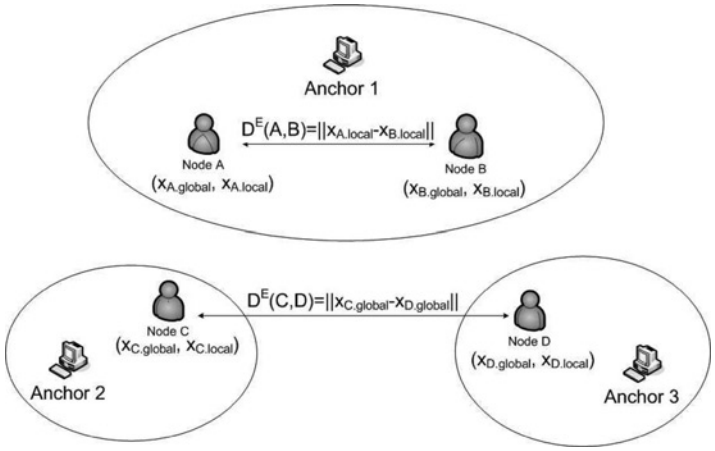

Figure 4 *Pharos overlay*

5 Performance evaluation

5.1 Experiment set-up

In our experiments, we compare Pharos with Vivaldi, with both the King and PlanetLab data sets. In Vivaldi, each node has 32 neighbours, which are randomly selected. Likewise, in Pharos, each node has 16 randomly selected neighbours in base overlay and 16 randomly selected neighbours in local cluster. This setting can guarantee that these two algorithms have the same overhead on the maintenance of neighbours. In each update interval, every Vivaldi node updates its NC once. It picks a random node among its neighbours, then pings this node and retrieves its NC. In Pharos, the local NC and global NC are updated by turns, that is, each Pharos node only updates either its global or local NC in each time interval. For a Pharos node, all its 32 neighbours can be regarded as global neighbours to update its global NC and the 16 neighbours in the same local cluster can be regarded as local neighbours to update its local NC. We run Vivaldi and Pharos both for 5000 update intervals and all nodes are persisted throughout the simulation. Therefore Vivaldi and Pharos have the same communication overhead, which guarantees that our improvement in the accuracy of NC is not acquired by introducing more measurement overheads. c_c and c_e in Vivaldi (also in each Vivaldi cluster in Pharos) is set to 0.25 as an empirical value in [12].

Similar to [3], we make the minimal assumptions about the placement of the anchors. We randomly select c anchors from the data sets. Guided by these anchors, the other nodes organise themselves into c proximity-based clusters. In our simulation, unless specified, c is set to 20. Thirty runs are performed on each data set and the average results are reported.

5.2 Results

As in Section 2, we use RE, the basic metric for NC prediction accuracy evaluation, as the metric to evaluate the performance of Pharos and Vivaldi. We vary the dimension of the Euclidean space from 3 to 12. Fig. 5 shows the comparison of average relative error between Pharos and Vivaldi. Pharos outperforms Vivaldi a lot in both data sets. Pharos can reduce the average relative error from Vivaldi by 45–55% in PlanetLab data set and 23–35% in King data set.

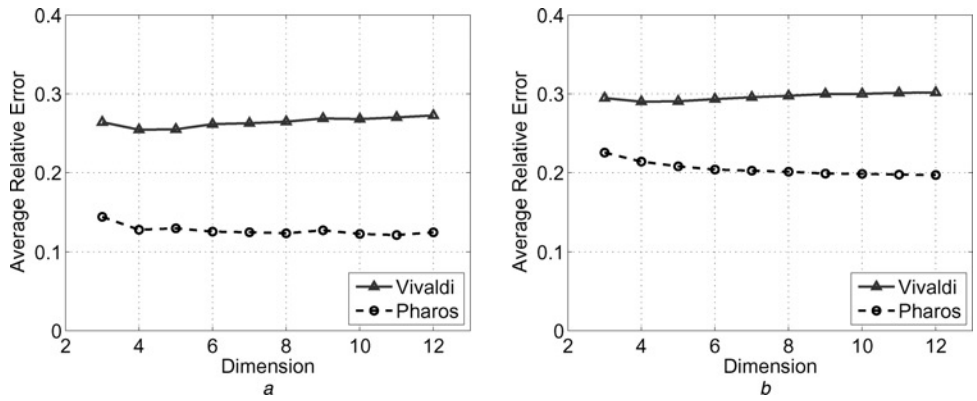

Figure 5 *Average relative error*

a PlanetLab
b King

IET Commun., 2009, Vol. 3, Iss. 4, pp. 539–548
doi: 10.1049/iet-com.2008.0187

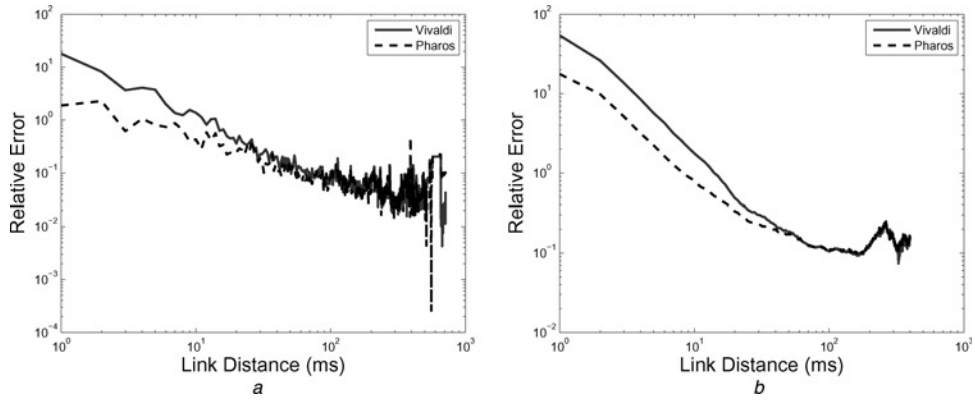

Figure 6 *Relationship between range of distance and relative error*

a PlanetLab
b King

To study the impact of the distance on prediction error, we set the dimension of the Euclidean as 7 and plot Fig. 6 to demonstrate the comparison of the relative prediction error for links of various distances between Pharos and Vivaldi. Pharos improves the prediction accuracy mainly for short links while achieving almost the same prediction accuracy with Vivaldi for medium and long links.

To study the impact of the density of the anchors on prediction error, we set the dimension of the Euclidean as 7 and use Table 1 to demonstrate the comparison of the average relative error with different number of anchors. We set the number of anchors as 10, 15, 20, 25, 30, respectively. From Table. 1, we can see that in both PlanetLab data set and King data set, the average relative error of Pharos varies only a little with different number of anchors. Therefore Pharos is insensitive to the number of anchors. This feature guarantees the flexibility in the deployment of anchors.

6 Performance in overlay multicast

6.1 Algorithms

NC plays an important role in scalable construction of overlay multicast tree. To study the performance of NC in overlay multicast, Zhang *et al.* [2] use three algorithms for overlay tree construction: minimum spanning tree (MST), modified ESM (a modified version from the typical ESM [28] Protocol) and LGK [29]. They point out that these three algorithms can capture the two essential building blocks of most overlay multicast protocols, namely shortest link selection and proximity-based clustering.

Table 1 Average relative error of Pharos under different number of anchors

	Anchors				
Data set	10	15	20	25	30
PlanetLab	0.12	0.12	0.13	0.13	0.13
King	0.22	0.21	0.20	0.20	0.20

For the construction of MST, Prim Algorithm is used. In modified ESM, the parent selection algorithm is different from that in [28]. In modified ESM, a new node selects the node with the smallest latency to itself in a randomly sampled partial list (30 nodes) of on-tree nodes; the link capacity and node degree are not considered in this simplified version. A location-guided k-ary (LGK) tree is constructed as follows: (1) the root node selects the k nodes with the smallest latencies to itself as its children nodes; (2) group the rest of the nodes to these k children nodes according to geometric proximity. As a result, each of the k children nodes become the root of a sub-tree. The multicast tree is formed, as each subtree repeats the two steps of child selection and clustering. It has been shown that $k = 2$ gives the best trade-off between the delivery delay and overhead of the multicast tree [2]. Thus, we will set k as 2 for our evaluation.

In all these three algorithms, we use the predicted distances, which are calculated by NC (Pharos or Vivaldi) for the overlay tree construction. Our simulation result will show the difference between Vivaldi and Pharos in overlay multicast.

6.2 Metrics

We use the same performance evaluation metrics as [2] for the above three algorithms. Tree cost is used to evaluate the NC-based overlay tree quality of the MST and modified ESM algorithms. Tree cost is defined as the sum of the latencies (measured latencies) over all links of the overlay multicast tree. Because LGK aims to optimise the delivery delay instead of the overall cost of the tree, delay stretch is used to evaluate the efficiency of the NC-based LGK Tree. Delay stretch is defined as the ratio of the latency on the overlay multicast tree and the delay of the direct unicast path between the root and a tree node.

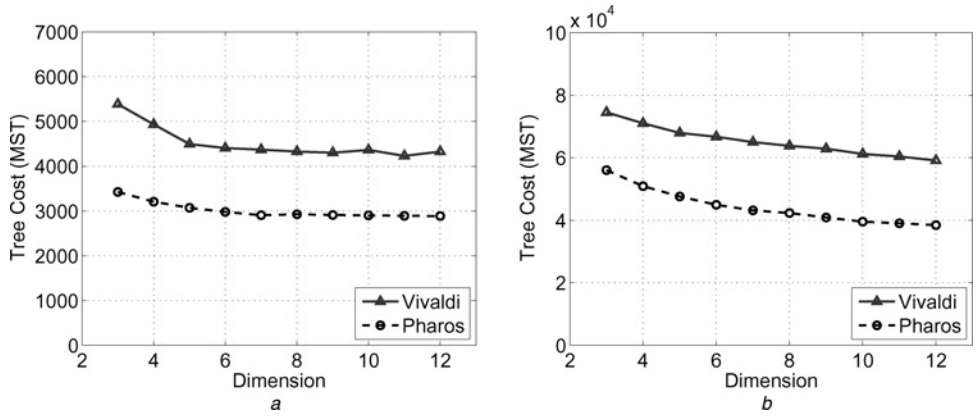

Figure 7 *Tree cost (MST)*

a PlanetLab
b King

6.3 Results

We vary the dimension of the Euclidean space from 3 to 12 to evaluate the performance of Pharos and Vivaldi in overlay multicast. In Fig. 7, it can be seen that Pharos performs much better than Vivaldi in MST, reducing the tree cost from Vivaldi by 32–36% in PlanetLab data set and 25–35% in King data set. In Fig. 8, we can also see that Pharos outperforms Vivaldi a lot in modified ESM, reducing the tree cost from Vivaldi by 9–17% in PlanetLab data set and 14–18% in King data set. In Fig. 9, we can see that Pharos still performs better than Vivaldi in LGK, reducing the average stretch from Vivaldi by 9–20% in PlanetLab data set and 12–21% in King data set.

7 Performance in server selection

7.1 Metrics

The replication of content over the Internet highlights the significance of the server selection problem. When we receive a list of multiple severs, we have to choose one of them to contact. Server load and RTT from the client are two critical parameters. As in [3], we focus on the latency parameter. In other words, the server closest to the client is defined as a good server.

We use the stretch to evaluate the performance of NC-based server selection. The definition of the stretch [2] is as follows

$$\text{stretch} = \frac{\text{latency-to-selected-server}}{\text{latency-to-true-closest-server}} \tag{6}$$

Smaller stretch indicates better performance in server selection.

The number of servers was set to 30 for PlanetLab data set and 200 for King data set. These servers are chosen randomly from the corresponding data set.

7.2 Results

We vary the dimension of the Euclidean space from 3 to 12 to evaluate the performance of Pharos and Vivaldi in closest

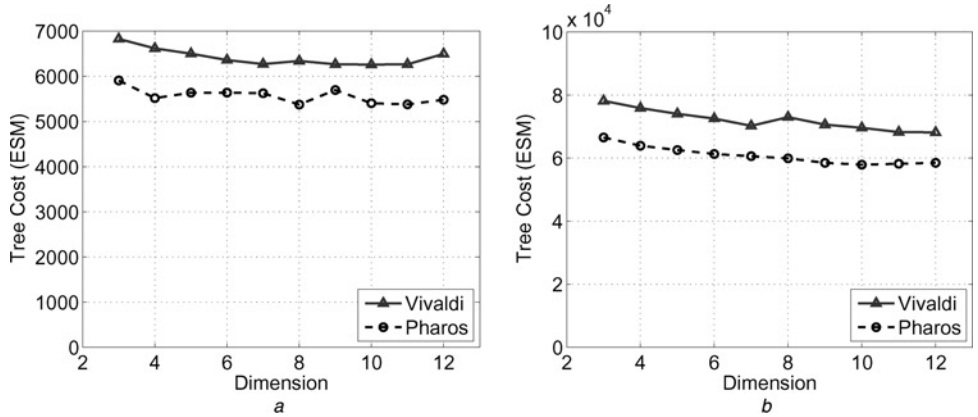

Figure 8 *Tree cost (ESM)*

a PlanetLab
b King

IET Commun., 2009, Vol. 3, Iss. 4, pp. 539–548
doi: 10.1049/iet-com.2008.0187

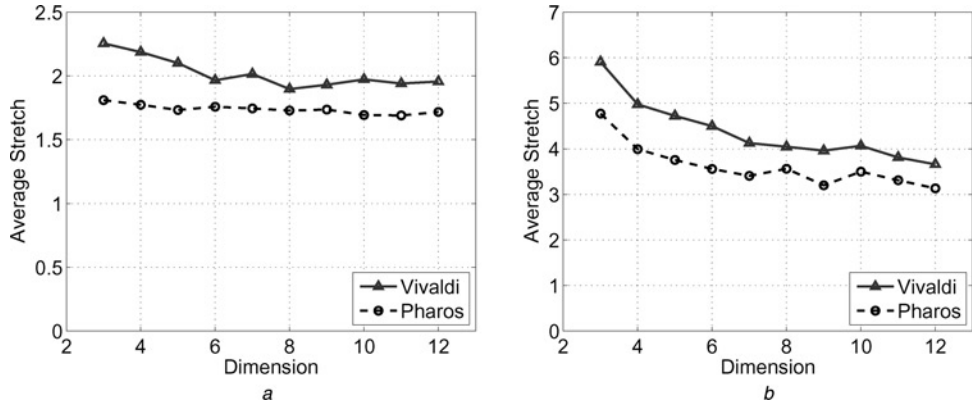

Figure 9 *Stretch (LGK)*

a PlanetLab
b King

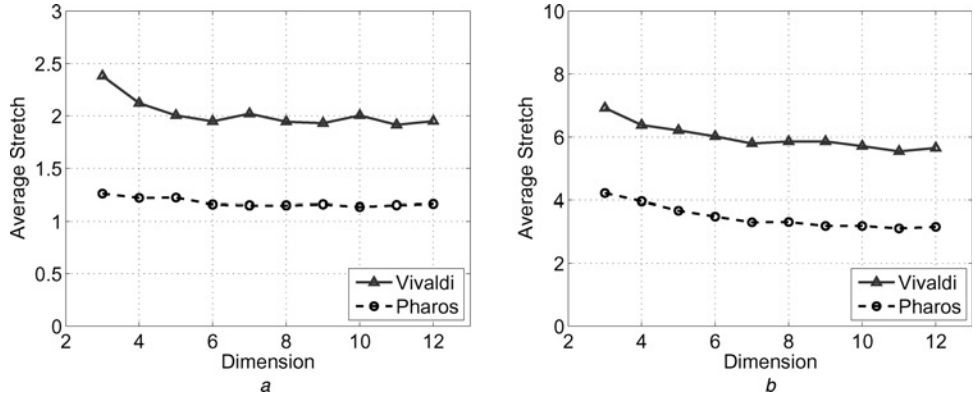

Figure 10 *Stretch in server selection using prediction*

a PlanetLab
b King

server selection. In Fig. 10, we can see that Pharos performs much better than Vivaldi in both PlanetLab data set and King data set. In PlanetLab data set, Pharos can reduce the average stretch from Vivaldi by 39–47%. In King data set, Pharos can reduce the average relative error from Vivaldi by 38–46%. In other words, Pharos improves the quality of the closest neighbour selection a lot when compared with Vivaldi.

8 Conclusion

In this paper, we study the causes of the prediction error for a representative simulation-based network coordination system, Vivaldi, and find out that the range of distance of peers has non-trivial impact on the performance of the system. We propose a multi-set coordinates scheme called Pharos to address this issue. Our contribution is 2-fold. (1) We analyse the distribution of the relative error of a representative SBA system, Vivaldi, and find out the relationship between the range of distance of peers and the prediction error. (2) We propose Pharos, a fully decentralised and hierarchical network coordinate system, to improve the accuracy of Internet distance prediction. We evaluate Pharos system with real Internet measurement traces. The extensive results of both typical NC metric and Internet applications perspective show that Pharos achieves better performance than Vivaldi, a representative distributed NC system.

To further evaluate the practicality of Pharos, we currently focus on deploying Pharos on Internet and developing some real applications based on this NC system.

9 Acknowledgments

This work is supported by the National Basic Research Program of China (No. 2007CB310806) and the National Science Foundation of China (Nos. 60473087, 60703052, 60850003). The authors thank Mr. Ang Li from Duke University for his comments and suggestions. This paper was presented in part at the IEEE GLOBECOM 2007, Washington, DC, USA, November 2007 [30].

10 References

[1] GUMMADI K.P., SAROIU S., GRIBBLE S.D.: 'King: estimating latency between arbitrary internet end hosts'. Proc. ACM SIGCOMM IMW, November 2002

[2] ZHANG R.M., TANG C.Q., HU Y.C., FAHMY S., LIN X.: 'Impact of the inaccuracy of distance prediction algorithms on internet applications: an analytical and comparative study'. Proc. IEEE INFOCOM, May 2006

[3] RATNASAMY S., HANDLEY M., KARP R., SHENKER S.: 'Topologically-aware overlay construction and server selection'. Proc. IEEE INFOCOM, June 2002

[4] PIETZUCH P., LEDLIE J.: 'Network-aware operator placement for stream-processing systems'. Proc. ICDE, 2006

[5] Azureus Bittorrent, http://azureus.sourceforge.net/, accessed August 2008

[6] ABRAHAM I., MALKHI D.: 'Compact routing on euclidian metrics'. Proc. PODC, 2004

[7] LEDLIE J., GARDNER P., SELTZER M.: 'Network coordinates in the wild'. Proc. NSDI, April 2007

[8] PIETZUCH P., LEDLIE J., MITZENMACHER M., SELTZER M.: 'Network-aware overlays with network coordinates'. Proc. IWDDS, July 2006

[9] NG T.S.E., ZHANG H.: 'Predicting internet network distance with coordinates-based approaches'. Proc. INFOCOM, June 2002

[10] PIAS M., CROWCROFT J., WILBUR S., HARRIS T., BHATTI S.: 'Lighthouses for scalable distributed location'. Proc. IPTPS, February 2003

[11] MAO Y., SAUL L., SMITH J.M.: 'IDES: an internet distance estimation service for large networks', *IEEE J. Selected Areas Commun.* (JSAC), Special Issue on Sampling the Internet, Techniques and Applications, 2006, **24**, (12), pp. 2273–2284

[12] DABEK F., COX R., KAASHOEK F., MORRIS R.: 'Vivaldi: a decentralized network coordinate system'. Proc. ACM SIGCOMM, August 2004

[13] SHAVITT Y., TANKEL T.: 'Big-bang simulation for embedding network distances in euclidean space'. Proc. IEEE INFOCOM, April 2003

[14] ZHANG R., HU Y.C., LIN X., FAHMY S.: 'A hierarchical approach to internet distance prediction'. Proc. IEEE ICDCS, July 2006

[15] LUA E.K., GRIFFIN T., PIAS M., ZHENG H., CROWCROFT J.: 'On the accuracy of embeddings for internet coordinate systems'. Proc. ACM IMC, October 2005

[16] TANG L.Y., CROVELLA M.: 'Geometric exploration of the landmark selection problem'. Proc. PAM, April 2004

[17] TANG L.Y., CROVELLA M.: 'Virtual landmarks for the internet'. Proc. ACM IMC, October 2003

[18] LIM H., HOU J.C., CHOI C.: 'Constructing internet coordinate system based on delay measurement'. Proc. ACM IMC, October 2003

[19] CHEN Y., ZHAO G.Y., LI A., DENG B.X., LI X.: 'Myth: an accurate and scalable network coordinate system under high node churn rate'. Proc. 15th IEEE Int. Conf. on Networks (ICON07), Adelaide, Australia, November 2007

[20] KAAFAR M.A., MATHY L., TURLETTI T., DABBOUS W.: 'Virtual networks under attack: disrupting internet coordinate systems'. Proc. ACM CoNext, December 2006

[21] KAAFAR M.A., MATHY L., BARAKAT C., SALAMATIAN K., TURLETTI T., DABBOUS W.: 'Securing internet coordinate embedding systems'. Proc. ACM SIGCOMM, August 2007

[22] RHEA S., GEELS D., ROSCOE T., KUBIATOWICZ J.: 'Handling churn in a DHT'. Proc. USENIX Annual Technical Conf., June 2004

[23] PlanetLab, http://www.planet-lab.org/, accessed August 2008

[24] NC Research Group at Harvard, http://www.eecs.harvard.edu/syrah/nc/, accessed August 2008

[25] CHU Y.H., GANJAM A., NG T.S.E., RAO S.G., SRIPANIDKULCHAI K., ZHAN J., ZHANG H.: 'Early deployment experience with an overlay based internet broadcasting system'. Prof. USENIX Annual Technical Conf., June 2004

[26] ZHANG B., NG T.S.E., NANDI A., RIEDI R., DRUSCHEL P., WANG G.H.: 'Measurement-based analysis, modeling, and synthesis of the internet delay space'. Proc. ACM IMC, October 2006

[27] Route Views Project, http://www.routeviews.org/, accessed August 2008

[28] CHU Y.H., GANJAM A., NG T.S.E., RAO S.G., SRIPANIDKULCHAI K., ZHAN J., ZHANG H.: 'Early experience with an internet broadcast system based on overlay multicast'. Proc. USENIX, June–July 2004

[29] CHEN K., NAHRSTEDT K.: 'Effective location-guided tree construction algorithms for small group multicast in MANET'. Proc. IEEE INFOCOM, June 2002

[30] CHEN Y., XIONG Y.Q., SHI X.H., DENG B.X., LI X.: 'Pharos: a decentralized and hierarchical network coordinate system for internet distance prediction'. Proc. IEEE GLOBECOM, November 2007

doi: 10.1049/iet-com.2008.0187