Embedding Network Coordinates into the Heart of Distributed Hash Tables

Toshinori Kojima Masato Asahara Kenji Kono Ai Hayakawa
Department of Information and Computer Science, Keio University
3-14-1 Hiyoshi, Kohoku-ku, Yokohama, Japan
E-mail: exapeer@sslab.ics.keio.ac.jp

Abstract

Network coordinates (NCs) construct a logical space which enables efficient and accurate estimation of network latency. Although many researchers have proposed NC-based strategies to reduce the lookup latency of distributed hash tables (DHTs), these strategies are limited in the improvement of the lookup latency; the nearest node to which a query should be forwarded is not always included in the consideration scope of a node. This is because conventional DHTs assign node IDs independent of the underlying physical network. In this paper, we propose an NC-based method of constructing a topology-aware DHT by Proximity Identifier Selection strategy (PIS/NC). PIS/NC assigns an ID to each node based on NC of the node. This paper presents Canary, a PIS/NC-based CAN whose d-dimensional logical space corresponds to that of Vivaldi. Our simulation results suggest that PIS/NC has the possibility of dramatically improving the lookup latency of DHTs. Whereas DHash++ is only able to reduce the median lookup latency by 15% of the original Chord, Canary reduces it by 70% of the original CAN.

1. Introduction

Network coordinates (NCs) are emerging technologies that allow us to estimate network latency on the Internet without explicitly communicating with target nodes. NCs map the geographical relationship among all nodes participating in the system on a virtual d-dimensional coordinate space. Since the Euclidean distance between any pair of coordinates approximates network latency between the corresponding nodes, each node estimates network latency to other nodes without adding an extra load to the network. As Ledlie et al. showed using Azureus BitTorrent clients [1], NCs work greatly on the real Internet [5].

There is still a need for more research on how to apply NCs to peer-to-peer (P2P) applications, such as distributed hash tables (DHTs). Although some studies which improve the lookup efficiency of DHTs using latency prediction by NCs have been conducted [3, 8], they do not make maximum use of the properties of NCs. In these studies, NCs are used for *Proximity Neighbor Selection (PNS)* or *Proximity Route Selection (PRS)* [4]. These DHTs just add optional information, the latency predicted by NCs, aside from ID on the DHTs' logical key spaces, and thus the benefit of NCs is limited by the flexibility of neighbor selection or route selection of DHTs. Therefore, even if there is a node very close to the destination node on the physical network, the close node may not be included in the forwarding candidates of the source node when they are far away on the logical space.

In this paper, we explore the possibility of using NCs for *Proximity Identifier Selection (PIS)* [4]. PIS is a fundamentally different strategy from PNS and PRS. The essence of PIS is that the nodes which are near on the underlay network are close to each other on the overlay network, too. As shown in our experiments, PIS with NCs (PIS/NC) has a greater potential for the lookup latency improvement than the other restricted strategies such as PNS and PRS though it poses several challenges. To make traditional DHTs into PIS/NC-based ones, we embed NCs directly into the *heart* of the DHT structure. In other words, we incorporate the logical space of NCs and that of a DHT. As a result, the DHT provides a logical key space which approximates the physical network topology.

To demonstrate the efficiency of PIS/NC, we design *Canary*, which combines CAN [6] and Vivaldi [2]. In Canary, each node is assigned a coordinate calculated from Vivaldi and these coordinates are in turn used to build CAN. Thus, the topological relationship between nodes is inherently embedded in the DHT structure of Canary.

Simulation results using Transit-Stub model [12] demonstrate that the lookup latency of Canary is the lowest among all prepared DHTs. Compared to original CAN, Canary reduces the median lookup latency by 70%. The reduction ratio is larger than that of DHash++ [3] by 55 points.

2. Design Challenges and Related Work

2.1. Design Challenges

Many researchers have studied strategies to improve the lookup latency of DHTs. Gummadi et al. [4] analyzed the effect of the overlay geometries used in various DHTs and classified the strategies for the lookup latency improvement with underlay information into three categories. According to [4], most of the strategies are categorized into two types of them, *PNS* or *PRS*.

- *Proximity Neighbor Selection* (PNS) chooses the nodes to be kept in a node's routing table according to the metrics based on the underlay information.

- *Proximity Route Selection* (PRS) considers the underlay metrics when a node chooses a forwarding node from its routing table to send a query to a particular destination.

Although PNS and PRS improve the lookup latency to a certain level of effect, unfortunately, they have inevitable limitations of the improvement of the lookup latency. In PNS- or PRS-based DHTs, the nearest node to which a query should be forwarded is not always included in the consideration scope of a node because the limitation of the candidates for each routing table entry (in the case of PNS) or the limitation of the routing protocol of the DHT (in the case of PRS).

To clear up the improvement limitations of the lookup latency, we focus on another strategy, *PIS*.

- *Proximity Identifier Selection* (PIS) chooses nodes' identifiers so that they reflect their positions on the underlay network.

PIS builds a DHT whose overlay topology is correlated to the underlay topology. In PIS-based DHTs, if a node is close to the destination on the overlay, it is also close to the destination on the underlay. Therefore, it can improve the lookup latency more than PNS and PRS.

In this paper, we explore a PIS-based method to improve the lookup latency in DHTs. Of course, as many researchers have concerned, PIS strategy poses some challenges. However, PIS has the possibility of improving the lookup latency dramatically. In fact, as described in Section 5, the reduction ratio of the lookup latency on a DHT with our proposal — a PIS-based method that constructs DHTs based on the logical space of NCs — is up to 70%, while that by a PNS-based DHT is 15%.

In what follows in this section, we discuss existing strategies by classifying into two categories; PNS and PRS, and conventional PIS.

2.2. PNS and PRS

Pastry [10] is one of traditional topology-aware DHTs. In Pastry, each node conducts PNS when it initializes and maintains its routing table. As a result, all routing table entries refer to a node that is near the present node on the physical network among all live nodes with a prefix appropriate for the entry. However, the improvement of lookup latency is limited due to the limit of selecting a candidate node for each row; a prefix of a candidate must be appropriate for the entry. Thus, an entry of a routing table will not always refer to the closest node.

DHash++ [3] is a representative DHT for improving the lookup latency with NCs. DHash++, which is based on Chord [11], takes PNS with Vivaldi [2]. In DHash++, the i-th Chord finger table entry of the node with ID a refers to the node with the lowest latency, estimated by Vivaldi coordinates, of up to the first x nodes in the ID space range $a + 2^i$ to $a + 2^{i+1}$. Thus, average lookup latency per hop becomes lower than that of original Chord. However, the improvement of lookup latency is limited due to the limit of the number of candidate nodes selected for each finger table entry; an ID of a candidate for the i-th entry must be within the ID space range $a + 2^i$ to $a + 2^{i+1}$. Thus, DHash++ will forward a query on a detour path, especially when it forwards a query to the lower row's entry of a finger table.

Rhea et al. [8] take PRS with NCs to improve the lookup efficiency of Bamboo [9], which is a customized Pastry to improve tolerance to churn. Although PRS-based Bamboo with NCs reduces the lookup latency, it still has the latency improvement limitations of Pastry as described above.

2.3. Conventional PIS

SAT-Match [7] is a heuristic strategy for matching overlay with underlying networks. SAT-Match gradually changes the overlay network to the one closer to the physical network by repeatedly measuring the latency between nearby nodes. In SAT-Match, each participating node periodically communicates with nodes which are reachable within small k hops and measures their latency. If there is a node which is close enough to itself on the underlying network, it "jumps" to that node's domain. "Jump" means that a node leaves its domain, and then joins another domain. This results in a small structural change. By repeating this, the overlay network of SAT-Match approximates the underlying network. However, since SAT-Match only considers heuristically the physical network topology in a limited scope on the overlay network, there is a limitation for the effect.

3. Proximity Identifier Selection with NCs

To clear up the improvement limitations of the lookup latency, we propose *Proximity Identifier Selection with Network Coordinates* (PIS/NC). PIS/NC evolves the traditional DHTs into PIS-based ones by embedding NCs *directly* into their structures. PIS/NC greatly reduces the lookup latency compared with not only the non-topology-aware DHTs, but also the existing topology-aware ones [3, 7, 8, 10].

PIS/NC is different from conventional PIS strategies such as SAT-Match; it constructs a logical ID space of a DHT based on the Euclidean space of NCs. PIS/NC determines a node ID on the basis of its network coordinate.

By doing this, a network coordinate space is directly embedded into the *heart* of a logical ID space of a DHT. Then, the overlay network topology of a PIS/NC-based DHT reflects the geographical relationship of the nodes; low-latency communicable nodes are close to each other on the logical ID space of a DHT, while high-latency communicable ones are distant from each other on the space. Thus, the forwarding path of a query on a PIS/NC-based DHT overlay is a good approximation of the path on the underlying network, i.e., the number of the query detouring on the physical network reduces significantly. PIS/NC essentially clears the latency improvement limitations and thus, it provides a great improvement of the lookup latency compared to PNS and PRS strategies even with NCs.

PIS/NC clears the improvement limitations of the lookup latency of conventional PIS strategies. Since these strategies determine a node ID in a heuristic way or simple node clustering, they often assign distant IDs to nearby nodes and similar IDs to distant nodes. Unlike them, PIS/NC provides a nearly-optimal topology-aware DHT. This is because PIS/NC builds a logical DHT space dependent upon an NC space.

4. Canary

To demonstrate the effectiveness of PIS/NC, we design *Canary*, which is a PIS/NC-based CAN. Canary uses Vivaldi coordinates when it constructs the overlay network of CAN. By doing this, Canary greatly reduces the lookup latency compared with original CAN.

To reduce such inefficient forwarding, Canary builds a PIS/NC-based overlay by embedding Vivaldi into CAN. In Canary, a node whose zone is adjacent is a closer node on the underlay. It reduces latency per hop and results in the significant reduction of the total lookup latency. Figure 1 illustrates an example of Canary[1]. In this example, a query is forwarded from A to B via F on the overlay. The forwarding path is significantly efficient from a point of view of the underlay.

Canary ensures that a node's neighbors are chosen from the nodes whose latency is low. To build a PIS/NC-based overlay, Canary *directly* maps Vivaldi coordinates of the nodes to d-dimensional Euclidean space of CAN. Since Canary assures that the coordinates of nodes are contained in each zone, the nodes managing adjacent zones are also close on the underlay network. Additionally, in Canary, each node adjusts its own zone according to the movement of Vivaldi coordinate.

5. Experiments

We conduct an experiment to demonstrate that PIS/NC reduces the lookup latency more than other strategies, i.e.,

[1]Although we let the dimensionality be two here to simplify, as described in Section 5, we set it to three in the experiment.

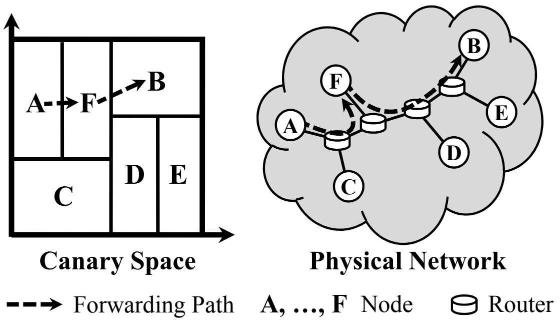


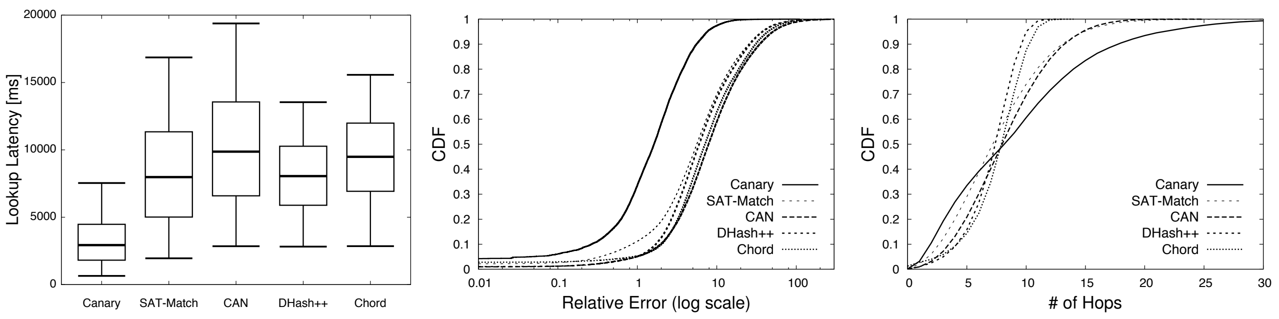
Figure 1. In Canary, query is forwarded on the efficient path. This is because the overlay network of Canary is integrated with physical network.

PNS strategies (DHash++ [3]) and conventional PIS strategies (SAT-Match [7]). We make each node send lookup queries to a random target and compare the degree of improvement by each strategy from two perspective of lookup latency (absolute value and relative error). In addition, we compare how they change the attributes of each base DHT from a perspective of the number of forwarding hops.

To evaluate and compare Canary and other DHTs with equal terms, we prepare Transit-Stub model [12] topology using GT-ITM. To generate the topology, we assigned the parameters as follows: 228 transit domains, 5 transit nodes per transit domain, 4 stub domains attached to each transit node, and 2 nodes in each stub domain. We select about 900 nodes from about 9,000 nodes. With this model, we tested each DHT operating 70,000 lookups. We set the dimensionality of the Euclidean coordinate space to three.

Figure 2 shows the experimental results. Figure 2 (a) shows that Canary reduces lookup latency compared with not only original CAN but also SAT-Match and DHash++. The lookup latency of Canary is the lowest among all prepared DHTs. The median latency of SAT-Match and DHash++ are reduced by 19% and 15% from their base DHTs; CAN and Chord. These reduction ratios are relatively low compared to that of Canary, which achieves greatly larger 70% reduction. Figure 2 (b) shows the cumulative distribution function (CDF) of the relative error. These graphs compare the lookup latency from another perspective, i.e., forwarding efficiency. They show that the relative error of Canary is less than those of other DHTs. This means that Canary forwards queries with a higher degree of efficiency than others. Compared with the median, it is 1.62 in Canary, while it is 5.84 in DHash++.

Figure 2 (c) shows the CDF of the number of hops. From these graphs, we can see that the distribution of forwarding hops of Canary is different from its base DHT, i.e., CAN compared with those of SAT-Match and DHash++. The reason seems to be non-uniform density of the node distribution. Although about half of lookups take more hops than



(a) Lookup Latency (b) Relative Error (c) Number of Hops

Figure 2. Results of experiment with Transit-Stub model. Canary reduces median of lookup latency by 70% compared with original CAN. This result is superior to those of SAT-Match and DHash++. Similarly, Canary is closest to ideal in terms of relative error as well. Although about half of lookups take more forwarding hops than other DHTs, maximum of lookup latency is the lowest.

original CAN due to dense regions, however, the latency per hop in such regions would be lowered because their coordinates are near each other. In fact, compared with the maximum lookup latency, the value of Canary is the lowest among the five DHTs, as shown in Figure 2 (a). Therefore, for most of P2P applications, the increase of hops does not really matter so much compared to the notable reduction of lookup latency.

6. Conclusion

We propose a novel strategy PIS/NC for the lookup latency improvement in distributed hash tables (DHTs). PIS/NC embeds network coordinates (NCs) into the heart of current DHTs. NCs are emerging technologies for accurately predicting communication latency between nodes. By taking PIS/NC, we can improve the performance of current DHTs. For our first attempt, we design Canary to demonstrate the possibility of reducing the lookup latency of current DHTs. By embedding Vivaldi coordinates in the mapping space of CAN, it integrates the physical link latency into the overlay network. Simulation results demonstrate that the lookup latency of Canary is the lowest among all prepared DHTs. Compared to original CAN, Canary reduces the median lookup latency by up to 70%. The results also show that the degree of improvement is larger than those of SAT-Match or DHash++. For future work, we plan to verify some technical challenges posed by PIS, e.g., load balancing or fault tolerance. In addition, we also plan to apply PIS/NC strategy to other traditional DHTs such as Chord, and evaluate the effect of them.

Acknowledgments

This work was partially supported by MEXT Grantin-Aid for Scientific Research on Priority Areas (New IT Infrastructure for the Information-explosion Era).

References

[1] Azureus BitTorrent Client. http://azureus.sourceforge.net/.
[2] F. Dabek, R. Cox, F. Kaashoek, and R. Morris. Vivaldi: A Decentralized Network Coordinate System. In *Proc. of ACM SIGCOMM*, pages 15–26, 2004.
[3] F. Dabek, J. Li, E. Sit, J. Robertson, M. F. Kaashoes, and R. Morris. Designing a DHT for low latency and high throughput. In *Proc. of USENIX Symp. on NSDI*, pages 85–98, 2004.
[4] K. Gummadi, R. Gummadi, S. Gribble, S. Ratnasamy, S. J. Shenker, and I. Stoica. The Impact of DHT Routing Geometry on Resilience and Proximity. In *Proc. of ACM SIGCOMM*, pages 381–394, 2003.
[5] J. Ledlie, P. Gardner, and M. Seltzer. Network Coordinates in the Wild. In *Proc. of USENIX Symp. on NSDI*, pages 299–311, 2007.
[6] S. Ratnasamy, P. Francis, M. Handley, R. M. Karp, and S. Shenker. A Scalable Content-Addressable Network. In *Proc. of ACM SIGCOMM*, pages 161–172, 2001.
[7] S. Ren, L. Guo, S. Jiang, and X. Zhang. SAT-Match: A Self-Adaptive Topology Matching Method to Achieve Low Lookup Latency in Structured P2P Overlay Networks. In *Proc. of IEEE IPDPS*, pages 83–91, 2004.
[8] S. Rhea, B.-G. Chun, J. Kubiatowicz, and S. Shenker. Fixing the Embarrassing Slowness of OpenDHT on PlanetLab. In *Proc. of WORLDS*, pages 25–30, 2005.
[9] S. Rhea, S. Rhea, D. Geels, D. Geels, T. Roscoe, T. Roscoe, J. Kubiatowicz, and J. Kubiatowicz. Handling Churn in a DHT. In *Proc. of USENIX*, 2004.
[10] A. I. T. Rowstron and P. Druschel. Pastry: Scalable Decentralized Object Location and Routing for Large-Scale Peer-to-Peer Systems. In *Proc. of IFIP/ACM Middleware*, pages 329–350, 2001.
[11] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan. Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications. In *Proc. of ACM SIGCOMM*, pages 149–160, 2001.
[12] E. W. Zegura, K. L. Calvert, and S. Bhattacharjee. How to Model an Internetwork. In *Proc. of IEEE INFOCOM*, pages 594–602, 1996.