Proximity Neighbor Selection using IP Prefix Matching in Kademlia-based Distributed Hash Table

Chayanon Sub-r-pa and Chotipat Pornavalai
Faculty of Information Technology
King Mongkut's Institute of Technology Ladkrabang
Bangkok, 10520 Thailand

***Abstract*- KAD is one of the most popular Peer-to-Peer (P2P) networks on the Internet. It uses Kademlia-based Distributed Hash Tables (DHTs) to form a tree-structured P2P network. Iterative routing algorithm is used on Kademlia to perform key-value DHT lookup operation based on XOR distance of peer identifiers, which are randomized. Therefore the lookup operation might increase the lookup latency, as well as introduce a lot of cross-network traffic to other Internet Service Providers (ISPs). In this paper, we propose a new efficient proximity neighbor selection in Kademlia-based DHT. The locality among the peers is simply justified based only on the IP prefix matching of peers' IP addresses. Unlike other existing approaches, there are no needs of local database of geographic information about peers, or modification of peer identification structure. Results from simulation on 2000 nodes network show that the proposed technique can reduce both the lookup latency and cross network traffic significantly, comparing to the traditional Kademlia lookup algorithm.**

I. INTRODUCTION

File sharing using peer-to-peer overlay communication is one of the most popular applications on the Internet nowadays. Their traffic and bandwidth usage is now a major faction of today's Internet traffic. Generally, there are two types of P2P file sharing P2P applications, namely "*unstructured P2P*" and "*structured P2P*" file sharing systems. Examples of the unstructured P2P are Napster [1], Guntella [2], KaZaa [3], and BitTorrent [4]. The peer selection and lookup are done in random on this type of P2P file sharing. On the other hands, with structured P2P network, the peer connectivity is organized in a structured manner such as ring or tree structure, and its indexing is performed using Distributed Hash Table. Examples of the structured P2P are Chord [5], CAN [6], Pastry [7], and Kademlia [8].

KAD is the most popular and largest Distribution Hash Tables (DHTs) P2P Network which is implemented on the Internet. It uses Kademlia-based DHT algorithm for indexing and lookup operation. Currently there are many file sharing applications running on KAD, such as eMule [9], and eDonkey2000 [10]. Normally we can see more than 1 million concurrent users on KAD. These users generate a lot of bandwidth usages on their ISP's network, and the links to other ISPs. Therefore ISPs are needed to upgrade their network infrastructure to maintain the quality of services to their customers.

In DHTs based system, peer identifier is randomized. The distance on the DHT peer-to-peer overlay network is calculated based on peer identifier. For example, on KAD the distance between two peers is the XOR of their peer identification. Therefore it has no relation to the distance on the real or underlay network topology. Without considering of locality in DHT, the underlying network lookup path between two peers can be significantly different from the path on the overlay network. Therefore the lookup latency on the overlay network could be quite high and affect the performance of the applications running on DHT [11].

It is true that the time needed to download a large file is mainly depended on the file size, whereas the query or lookup for the peer before starting download can be negligible. However, the lookup operation on the KAD which is iterative routing requires many queries and considerably increases the lookup latency [12]. There are many research works try to improve lookup performance and reduce lookup latency by considering network locality on DHT. For example, on Kademlia-based DHT, several techniques such as using parallel lookup, multiple replicas [13], cluster underlay metric [11], and modification of peer identifier to reflect its Autonomous System Numbers (ASNs) [11] were proposed.

This paper presents a new efficient way to form network locality among peers on Kademlia-based DHT network using modified Proximity Neighbors Selection (PNS) algorithm. We proposed underlay metric based on IP prefix matching of the peers' IP addresses, to proximate the underlying distance between the peers. Therefore, no need to lookup external database that contained geographic location information of each peer's IP addresses.

The main contributions of this paper are as follows:

- We found that the similarity of IP addresses (ip prefix matching) of the peers have some relations on the geographic location (ISP, country, continent) of the peers with considerably high precision.
- We showed that performing IP prefix matching of the peers to estimate the proximity of peers (neighbors) on Kademlia-based DHT network could reduce lookup latency and cross-traffic to other ISPs significantly.

The rest of this paper is organized as follows. First we give the background of Kademlia in Section II. Review of related works is in Section III. Then we present the detail of

proximity algorithm and the proposed IP underlay metric in Section IV and V respectively. Evaluation on its performance is in Section VI. We then conclude the paper in Section VII.

II. BACKGROUND OF KADEMLIA

Kademlia is the one of DHT system which presents an elegant distribute solution for deterministically mapping items to locations. Each Kademlia peer has an own unique identifier in 160-bit from hashing function (such as SHA-1). In the same way, content or value in system also have a 160-bit unique identifier so called "*key*". {*key*, *value*} pairs are stored on peer with its identifier "closest" to the key. The notation of closeness or distance between peers in the key space in Kademlia is based on XOR metric. The overview of routing on Kademlia is explained below [12].

For each order of distance each peer keeps a bucket with contact information which is a list of <*IP Address, UDP port, peer ID*> triples about *k* other peers. If the peer's identifier is $i = a_1a_2\ldots a_{160}$, then each bucket j, $j = 1\ldots160$, may contain up to *k* contacts with identifiers $b_1b_2\ldots b_{160}$ where $b_k = a_k$ for $1 \leq k < j$ and $b_j \neq a_j$. That is each peer has one bucket for all peers with the first identifier bit begin different from its own, another bucket for the first bit begin the same but the second bit different, and so on. *k* is to be chosen to provide resilience under peer turnover [12].

The operation to lookup the data items associated with a key $x \in \{0, 1\}^{160}$ in the DHT proceed as follows ("close" refers to the XOR metric) [12]:

1) *Step1:* The initiator of the lookup selects the *k* contacts from its routing table that are closest to *x*.
2) *Step2:* It then sends concurrent request messages to the closest contacts from this set that have not yet been queried. At most α message may be in transit at the same time.
3) *Step3:* If the receiver of a request message possesses a copy of the requested data item, it returns it to the initiator. Otherwise, it returns the *k* contacts from its own buckets that are closest to *x*.
4) *Step4:* If a reply message with the requested data item arrives at the initiator, the lookup terminates immediately.
5) *Step5:* If a reply message contains further contacts, the initiator inserts them into its set of the *k* closest contacts currently known, throwing away the most distant contacts and then continues at step 2.
6) *Step6:* If a timeout occurs, the lookup resumes step 2.
7) *Step7:* If no request message is in transit and all *k* closest contacts have been queried, the lookup terminates; there seems to be no data item that corresponds to the key *x*.

III. RELATED WORK

There are many research papers focus on how to keep locality in DHT lookup protocol. In [14], the authors presented and classified techniques that can be used to perform locality lookup on DHT network in to three categories as follows:

A. Proximity Neighbor Selection (PNS)

This technique applies when choosing new neighbors. The neighbors are selected based on the cost of the underlay metric. The neighbor peer that has low cost, which is low underlay metric, will be selected. In Kademlia, the list of neighbor is stored on the *k*-buckets. Therefore this technique tries to fill the *k*-bucket with the peers that are closest based on the underlay metric as many as possible. In [12], the authors propose PNS algorithm that use cluster underlay metric. The peers can use local database that map from IP address ranges to ISP, Region, Country, Continent, and World. However, in our proposed algorithm, we use only IP address prefix matching to predict whether peers are belong to the same ISP, Country, and Continent.

B. Proximity Routing Selection (PRS)

This technique applies to the routing algorithm and selects peers with the lowest cost in the routing table to perform queries. However, it is shown in [14] that PNS has better latency improvement than PRS. By combining PNS and PRS, it results in a small performance improvement than PNS alone. Therefore in this paper, we did not implement and evaluate PRS with the proposed IP underlay metric.

C. Proximity Identifier Selection (PIS) or Geographic Layout

Geographic layout is the way to add information about locality of peer in part of peer identifier. The LDHT [11] is classified in this category. In LDHT, the peer identifier is the combination of the mod value of ASN number and the random value. However, this technique may introduce load balancing problem on Kademlia network.

IV. PROXIMITY NEIGHBORS SELECTION

This paper applies only proximity neighbor selection (PNS) which aims to keep the least cost contacts in routing table. PNS on Kademlia-based DHT can be described into two strategies.

A. Replacement Strategy

Replacement strategy applies in Kademlia when bucket is full and all peers in this bucket are still remaining in network. In this case traditional Kademlia will reject the new coming peer. But when the strategy is applied, it will calculate and compare underlying cost between new coming peer and existing peer in bucket, if new coming peer has lower cost than existing peer, then the existing peer will be replaced with new coming peer. The detail about this strategy is presented with pseudo-code in the appendix. The underlying cost will be explained in section V.

B. Return Best k-Closest Strategy

With this strategy, peer will collect the list of known neighbors that are closest to key in term of XOR distance. Then, within this list, it will return *k*-closest (using underlay metric) contacts to the requested peer. Therefore the result of k-closest has lower cost than native mechanism. The detail

about this strategy is presented with pseudo-code in the appendix.

V. IP Underlay Metrics

In [12], "Cluster Underlay Metrics" is proposed to manage locality of traffics by using geographic information of peers. MaxMind [15] provides GeoIP database which contains geographic information about IP address ranges which is classified as continent, country, region, ISP and ASN. When peers need to calculate the underlay cost, PNS strategy determines the least cost is geographic distance between peers in same ISP (c=0) and increased by one for distance in same region (c=1), same country (c=2), same continent (c=3) and for peers with nothing in common at all (c=4). Table I summarize the underlay metric used in cluster PNS [12].

TABLE I
CALCULATION OF THE COST VALUE IN THE CLUSTER UNDERLAY METRIC

Underlay Metric	Peer located in same	Cost
Cluster	ISP Region Country Continent World	0 1 2 3 4

In "Cluster Underlay Metrics" approach, we need to obtain and search on the geographic information database, that constrain will increase the lookup time and may not be feasible for some peers. We try to classify cost in the same way of "Cluster underlay metric" without using geographic information database. Therefore we investigate whether how accurate about geographic information between two peers can be obtained using only IP address prefix matching.

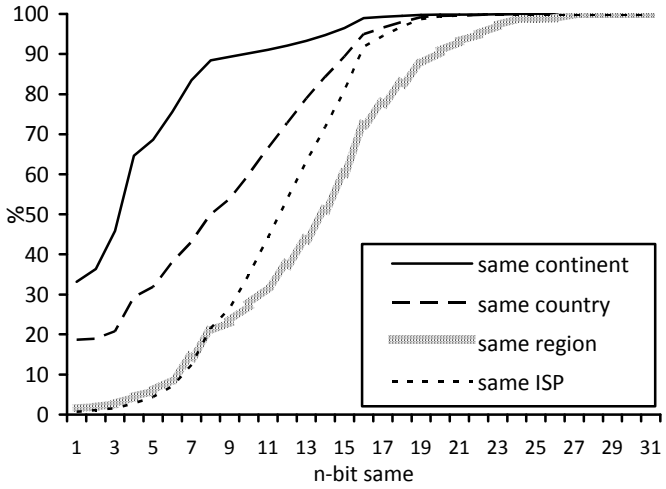

Figure 1.Matching n-bit prefix of IP address relation to geographic information

We random 100,000 IP addresses from the range of IP addresses from database provided by MaxMind GeoIP, and check IP prefix matching for all pairs. The result is then compared with the GeoIP database, and is shown in Figure 1. We found that 88% of the same 8-bit prefix IP addresses are stay in a same continent. With same 16-bit IP address prefix, 94.95% are in a same country, and 91.78% in a same ISP. However we found that the relation between IP address and region is not so strong. Also the geographic information of the ISP can be obtained with less number of bit prefix matching, so we exclude region from cost table. However, region relation still remains in our measurement.

IP address is native information in network that makes any peers can retrieve and calculate cost in IP underlay metrics. But this relation is only subset of geographic information database. Our heuristic selects least cost (c) by a same 16 bits IP address prefix or more ($c=0$). With same eight bit prefix or more ($c=1$) and for peers having nothing in common on the first 16 bits ($c=2$). We choose 8 and 16 bits because they are the critical points that give high precision of the geographic location, and their slopes are moving flatter. Our simple approach classifies cost calculation by IP address octets, and the calculated cost is shown in Table II.

TABLE II
CALCULATION OF THE COST VALUE IN THE IP UNDERLAY METRIC

Underlay Metric	IP address of peer in same	Cost
IP	16-bit prefix or more 8-bit – 15-bit prefix Less than 8-bit prefix	0 1 2

VI. Evaluation

For our experimental, we use PeerfactSim [16]: large scales P2P Simulation that support PNS algorithm and provide geographic information measurement dataset, that combine information from two internet measurement projects, CAIDA [17] and PingER [18]. This simulator can predict internet network distance using a technique called Global Network Positioning (GNP) [19]. This will give simulation result more reliable and near to real network.

A. Simulation Setup

The protocol parameters of Kademlia are set to k=20, b=2 and α=3. These parameters are chosen to be the same as in [reference to peer next door]. In order to evaluate the underlay metric that is close to reality, we use realistic geographic distribution of peer from KAD network [20].

In a measurement study of KAD network, Steiner et al observed a peer distribution. The observed results provide the distribute group by country in only 8-bit zone 0x5b prefix on KAD peer identifier call "zone crawl". They also observed that the distribution of 8-bit prefix is very near to the full 32 bits zone.

We build histogram of geographic distribution of the peers used in experiments by averaging histogram of 179 days from zone crawl. The size of network in our simulation is 2000 nodes with geographic distribution as shown in Figure 2. However dataset of IP address for simulation provided by PeerfactSim.KOM is not included IP addresses in South Korea, we ignore peer in this area and percentage of mission peer has been allocated to the remaining countries.

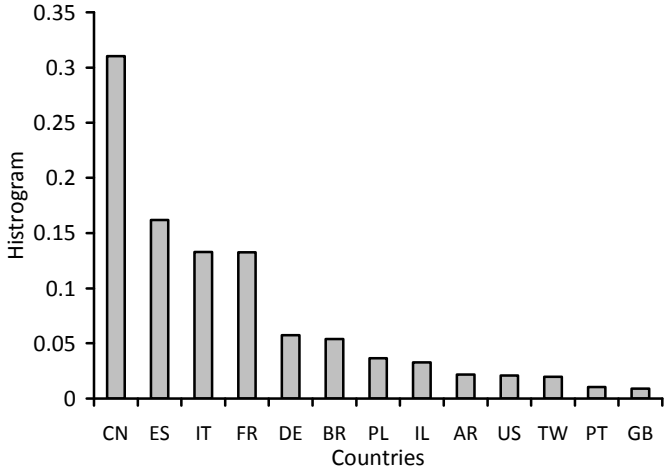

Figure 2. Histogram of geographic distribution peer seen on 2006/09/23 to 2007/03/20

B. *Simulation Result*

We complete our simulation with 99.9% success rate for key-value lookup. The measurement results are shown in term of successful lookup latency and number of messages used by lookup queries, as shown in Table III.

TABLE III
RESULT OF SIMULATION

	Lookup Latency (msec)	Number of Messages
Basic Kademlia	660.40	4.91
Cluster PNS	553.46	4.70
IP PNS	609.66	4.98

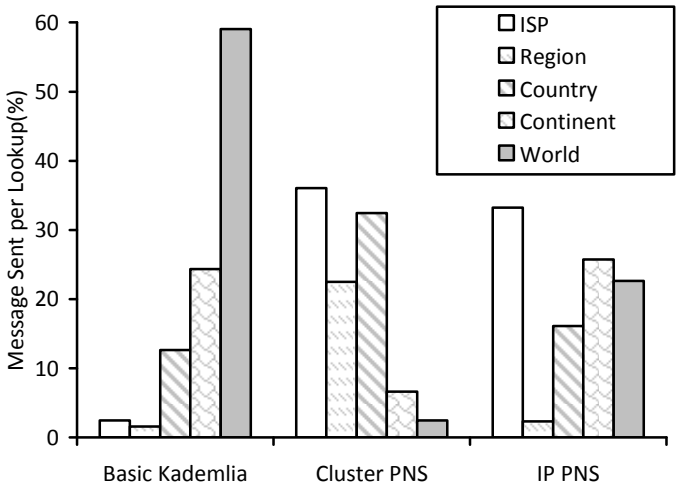

Figure 3. The distribution of messages sent during lookups.

The results show that the proposed PNS algorithm (IP PNS) that uses IP underlay metric has lower latency compared to the basic or traditional Kademlia. From Table III, the number of lookup messages required by IP PNS is just a little bit higher than Basic Kademlia and Cluster PNS. However, if we see the distribution of messages sent during lookups that is shown in Figure 3, more than 33% of lookup messages are within the same ISP, where it is only 3% on Basic Kademlia. Though Cluster PNS [12] is better than basic Kademlia and IP PNS, IP PNS is simpler. The IP address of the peer can be found in the bucket, so there is no extra overhead comparing with Cluster PNS which required local geographic information database. Another advantage of IP PNS is that prefix matching operation is very simple.

VII. CONCLUSION

The proposed IP underlay metric is very simple to implement with PNS algorithm. It has better performance in terms of lookup latency, number of lookup messages, and geographic information distribution of lookup messages, than traditional Kademlia lookup algorithm. With "Cluster Underlay Metric" we need geographic information from local or external database. By not using external information, we found that simply using "IP Prefix Matching", the performance is just slightly degraded comparing with Cluster PNS. But IP PNS will have much less overhead.

In the feature work, we will experiment in performance term of IP prefix matching algorithm compare with basic Kademlia and cluster underlay metrics. The performance term meant client's CPU workload, Replacement strategy frequency and effect from Replacement strategy.

APPENDIX

Proximity neighbor selection (PNS) in section IV can be explained with pseudo code below.

Algorithm 1 Replacement Strategy

```
01: procedure replacement(b, n) //bucket, new entry contact
02:    thisPeer = peer who use this algorithm
03:    mostContact = getMostCostContact(b) //get contact that
has most cost
04:    mostCost = costCalculate(thisPeer, candidate)
05:    newEntryCost = costCalculate(thisPeer, n)
06:    if mostCost > newEntryCost then
07:        bucket.remove(candidate)
08:        bucket.add(n)
09:    end if
10: end procedure
11: procedure getMostCostContact(b)
12:    mostCost = costCalculate(b[0])
13:     mostContact = bucket[0]  //get contact index = 0 in
bucket
14:    bucketSize = size of bucket
15:    for i = 1 to bucketSize loop
16:            thisCost = costCalculate(thisPeer, bucket[i])//get
contact index = 0 in bucket
17:       if mostCost < thisCost then
18:          mostCost = thisCost
19:          mostContact = bucket[i]
20:       end if
```

```
21:    end loop
22:    return mostContact
23:end procedure
24:procedure costCalculate(o, c) //originate peer, contact peer
25:    return from cost table
26:end procedure
```

Algorithm 2 Return best k closest

```
01:procedure returnBest(key, o, k) //target key, originate nodeID, k from Kademlia parameter
02:    resultList = empty list of contact//prepare result list
03:    closestList = empty list of contact//prepare best contact with XOR
04:    d = peerID XOR key//calculate distant between this peer and key
05:    for each bucket in this peer loop
06:        bSize = size of bucket
07:        for i = 0 to bSize loop
08:                distant = bucket[i] XOR key //calculate distant between contact index i and key
09:                if distant <= d then //if contact is closest to key than this peer
10:                closestList.add(bucket[i]) //insert this contact to closestList
11:            end if
12:        end loop
13:    end loop
14:    sort closestList order by distant
15:    maxCost = maximum cost in calculate cost table
16:    currentCost = 0
17:    kSize = size of resultList  //size of k begin from 0
18:    while currentCost <= maxCost and kSize <= k do
19:        cSize = size of closestList
20:        for i = 0 to cSize and kSize < k loop
21:            thisCost = costCalculate(o, closeList[i])
22:            if thisCost = currentCost then
23:                resultList.add(clossList[i])
24:            end if
25:        end loop
26:        currentCost = currentCost + 1
27:        kSize = size of resultList
28:    end while
29:    return resultList
30:end procedure
```

REFERENCES

[1] Napster online music file sharing service: http://free.napster.com/

[2] Gnutella file sharing and distribution network: http://rfc-gnutella.sourceforge.net/

[3] Kazaa file sharing service: http://www.kazaa.com/

[4] BitTorrent open source file-sharing application,: http://www.bittorrent.com/

[5] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, “Chord: A scalable peer-to-peer lookup protocol for internet applications, “ IEEE/ACM Transactions on Networking val. 11, 2003

[6] Ratnasamy, s., Francis, P., Handley, M., Karp, R., and Shenker, S. A, "Scalable content-addressable network, “ In Proc. ACM SIGCOMM , 2001.

[7] Antony Rowstron and Peter Druschel, “Pastry: Scalable, decentralized object location and routing for large-scale peer-to-peer systems, “ in Proc. IFIP/ACM International Conference on Distributed System Platforms, 2001.

[8] Petar Maymounkov and David Mazieres, “Kademlia: A Peer-to-peer Information System Based on the XOR Metric,” in International Workshop on Peer-to-Peer Systems, 2002.

[9] eMule: http://www.emule-project.net

[10] eDonkey2000: http://www.edonkey.co.nr/

[11] Weiyu Wu, Yang Chen, Xinyi Zhang, Xiaohui Shi, Lin Cong, Beixing Deng, Xing Li, “LDHT: Locality-aware Distributed Hash Tables,” Proceedings of the ICOIN 2008 Information Networking, 2008.

[12] Sebastian Kaune, Tobias Lauinger, Aleksandra Kovacevic, Konstantin Pussep, “Embracing the Peer Next Door: Proximity in Kademlia,” Peer-to-Peer Computing, IEEE International Conference, 2008

[13] Daniel Stutzbach, Reza Rejaie, “Improving Lookup Performance over a Widely-Deployed DHT,” Proceedings of the 25th IEEE International Conference on Computer Communications. Proceedings INFOCOM, 2006

[14] Krishna P. Gummadi, Ramakrishna Gummadi, Steven D. Gribble, Sylvia Ratnasamy, Scott Shenker, and Ion Stoica., “The Impact of DHT Routing Geometry on Resilience and Proximity,” Proceedings of the ACM SIGCOMM, 2003.

[15] Max Mind Geolocation Technology. http://www.maxmind.com

[16] PeerfactSim.KOM: A Large Scale Simulation Framework for Peer-to-Peer Systems, http://peerfact.kom.e-technik.tu-darmstadt.de/

[17] Cooperative Association for Internet Data Analysis(CAIDA). Macroscopic TopologyProject, http://www.caida.org/

[18] PingER (Ping End-to-end Reporting), http://www-iepm.slac.stanford.edu/pinger/

[19] T. S. Eugene Ng and Hui Zhang, “Towards Global Network Positioning,” Proceedings of the First ACM SIGCOMM Workshop on Internet Measurement, 2001

[20] Moritz Steiner, Taoufik En-Najjary, and Ernst W. Biersack, “A global view of KAD”, Proceedings of Internet Measurement Conference (IMC), October 2007, San Diego, USA