[KLKP08] Sebastian Kaune, Tobias Lauinger, Aleksandra Kovacevic, Konstantin Pussep: Embracing the Peer Next Door: Proximity in Kademlia.

In: 8th International Conference on Peer-to-Peer Computing 2008 (P2P '08), p. 343-350, IEEE Computer Society Press, September 2008. ISBN 978-0-7695-3318-6.

Embracing the Peer Next Door: Proximity in Kademlia

Sebastian Kaune, Tobias Lauinger, Aleksandra Kovačević, and Konstantin Pussep
Technische Universität Darmstadt
KOM – Multimedia Communications Lab
Merckstraße 25, 64283 Darmstadt, Germany
{kaune, lauinger, sandra, pussep}@kom.tu-darmstadt.de

Abstract

At present, the probability of selecting "the peer next door" as an overlay neighbour in Kademlia is fairly small. Prior research has been concerned with reducing the lookup latency by means of proximity neighbour and route selection, but focused on recursive routing algorithms.

This work leverages location data about peers and extends Kademlia's iterative routing algorithm to reduce cross-network traffic at the level of the distributed hash table. Evaluation with real-world measurement data gives evidence that locality of traffic tends to reduce lookup latencies as well. In turn, mechanisms that aim at reducing lookup latencies do not necessarily reduce cross-network traffic to the same extent.

1 Introduction

Peer-to-peer (P2P) accounts for more than 50 % of today's Internet traffic [8, 9] and represents a huge cost for Internet service providers (ISPs). Because many broadband users subscribe to flat rate tariffs, they stay online for long periods of time and have no incentive to limit their bandwidth consumption. For instance, Steiner et al. [19] report that 40 % of the peers in Kad have an online time of at least 5 hours per day; still 20 % are online for more than 10 hours, and some sessions last as long as 78 days.

These users generate a disproportionately high load on their ISP's network. ISPs view this as a threat to their business models as they are forced to upgrade their network infrastructure without making extra money from it.

Locality mechanisms attempt to alleviate this problem by keeping the largest possible fraction of P2P traffic inside an ISP's network. To this end, they take into account a peer's position in the underlay network instead of communicating with peers from arbitrary locations. In addition to reducing expensive cross-network traffic, this approach has clear benefits for users in that it considerably reduces the time required for file downloads [2], for instance.

Until now, locality algorithms have been applied at the content distribution level of P2P applications [2, 23]. This paper focuses on a complementary approach: DHT-level locality.

We consider the Kademlia [11] DHT due to its iterative lookup algorithm. (Iterative routing is used in currently deployed DHTs such as Kad [20] and Azureus [5].) In contrast to recursive routing algorithms, the initiator of an iterative lookup is in continuous control of the lookup process. When receiving candidates for the next hop, the initiator can examine their proximity attributes and then select the best peer from a set of peers considered reasonably equivalent in terms of the overlay (XOR) metric.

As there is a vast array of conceivable definitions of underlay proximity, we introduce an abstraction called *underlay metric*. For any pair of peers, an underlay metric permits the calculation of a cost value that expresses how desirable it is for the given two peers to communicate. This abstraction allows the selection of an underlay metric that best optimises routing for the specific needs of the respective application. For instance, previous work defined proximity in terms of round-trip time to reduce lookup latencies [16]. In this paper, we introduce an underlay metric that achieves locality of DHT traffic by preferentially routing to peers from the same ISP.

The iterative nature of Kademlia's lookup algorithm imposes limits on how the proximity attributes of a peer can be determined. When a large file is to be downloaded, the additional delay and traffic caused by querying a special node for the nearest peers is negligible. Yet this mechanism requires far too many queries and considerably increases the lookup latency when applied to iterative routing. Similar problems arise with those approaches to reducing lookup latencies that require that probe messages be sent to the peers in consideration. In order not to increase lookup latencies, only underlay metric implementations that return the cost value without time delay are suitable for iterative routing.

To the best of our knowledge, previous work on DHT proximity considered only recursive routing and had the lookup latency as the sole optimisation goal. The contribution of this work is as follows:

1. We extend the routing table and iterative routing algorithm of Kademlia to take into account information provided by an underlay metric. In doing so, the structure of Kademlia and its key properties related to the stability of the overlay are maintained.
2. We devise a Cluster underlay metric that queries a local copy of the MaxMind GeoIP database [10] for the ISP, region, country, and continent of peers. This information is then used to calculate a cost value that reflects geographic proximity. We also implement Vivaldi [4] for the class of network coordinate algorithms based on round-trip time observations.
3. We evaluate the proximity algorithms together with the underlay metrics in simulations with 10,000 peers. The scenario follows observations of the Kad network [20], and the underlay model uses real-world round-trip time measurement data [3]. Our results show that the Cluster underlay metric significantly reduces cross-network traffic and that lookup latencies tend to be reduced as well.

We begin by discussing related work in Section 2 and giving background on Kademlia in Section 3. Section 4 introduces our generic proximity algorithms and Section 5 shows how we use a novel Cluster underlay metric to achieve locality of traffic. Section 6 describes a measurement-based underlay model and the scenario used for the evaluation, the results of which we discuss in Section 7. Finally, Section 8 indicates future work and concludes this paper.

2 Related Work

There has been a substantial amount of work on reducing the end-to-end path latency in DHTs by considering neighbourhood proximity in recursive routing, e.g. [1, 6, 7, 15, 16, 24]. However, we are not aware of prior studies that contrast the applicability of underlay metrics in iterative routing (as it is being used in Kademlia).

Gummadi et al. [20] analyse the effect of the overlay geometries used in different DHTs on the flexibility in choosing neighbours and routes in the overlay network. They classify techniques that use information about the underlay into three categories, according to where the techniques are applied in the DHT:

- *Proximity Neighbour Selection (PNS)* selects the peers to be saved in a peer's routing table according to the cost computed by the underlay metric.
- *Proximity Route Selection (PRS)* applies to the routing algorithm and selects peers with a low cost in the routing process.
- *Proximity Identifier Selection (PIS)* refers to choosing peer identifiers to reflect the peers' positions in the underlay network. As this implies issues with load balancing, Gummadi et al. did not pursue this approach, and neither will we.

Focusing on recursive routing and assuming ideal conditions, Gummadi et al. provide a general overview of the performance that can be achieved with PNS and PRS, independent of the mechanisms that are used to implement them. Their analysis reveals that PNS offers a better latency improvement than PRS, whereas each of them can significantly improve the lookup latencies of a plain DHT. However, combining PNS and PRS results in only a small performance boost compared to PNS alone.

Rhea et al. [17] carry out an emulation-based analysis of several sampling techniques for PNS under peer turnover in the Bamboo DHT. These methods consist of discovering new entries for the routing table in different ways and taking round-trip time samples to determine the suitability of each of the discovered peers. While these methods can also be applied to PNS in Kademlia, we argue that in this framework, actively discovering new routing table entries is largely redundant: as routing in Kademlia is iterative, many potential candidate entries come in during normal operation. In contrast, these methods cannot be applied to PRS, as iterative routing prohibits taking round-trip time samples during the lookup to select the next hop. Furthermore, these approaches optimise routing for latency, but do not exercise the control needed to achieve locality of traffic.

3 Background on Kademlia

Each Kademlia peer owns a unique identifier $i \in \{0,1\}^{160}$. Distances in the overlay network are expressed in terms of the XOR metric. They are roughly equivalent to the length of the common prefix of two identifiers.

3.1 Routing table

For each order of distance, each peer keeps a *bucket* with contact information about k other peers. (This information is also called a *contact*.) If the peer's identifier is $i = a_1 a_2 \cdots a_{160}$, then each bucket j, $j = 1..160$, may contain up to k contacts with identifiers $b_1 b_2 \cdots b_{160}$ 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 being different from its own, another bucket for the first bit being the same but the second bit different, and so on. k is to be chosen to provide resilience under peer turnover.

3.2 Routing algorithm

The operation to look up the data item associated with a key $c \in \{0,1\}^{160}$ in the DHT proceeds as follows ("close" refers to the XOR metric):

1. The initiator of the lookup selects the k contacts from its routing table that are closest to c.

2. It then sends concurrent request messages to the closest contacts from this set that have not yet been queried. At most α messages may be in transit at the same time.

3. 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 routing table that are closest to c.

4. If a reply message with the requested data item arrives at the initiator, the lookup terminates immediately.

5. 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. If a timeout occurs, the lookup resumes at step 2.

7. 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 c.

4 Proximity Algorithms for Kademlia

This section briefly outlines our PNS and PRS algorithms for Kademlia. Their main goal is to improve routing with respect to the underlay without making structural changes to the overlay. In particular, all key characteristics of Kademlia, such as stability in the presence of peer turnover, have to be maintained. To this end, we define equivalence relations on the overlay metric, and use an underlay metric to select the best peer from a set of equivalent contacts.

The proximity algorithms described in this section work with any underlay metric that complies with the definition from Section 1. That is, the algorithms are implemented to use an abstract underlay metric interface that consists of a function `int cost(sender, receiver)`. The value returned by this function estimates the expense of a message sent from `sender` to `receiver` and must be calculated without time delay. Specific implementations of this interface will be discussed in Section 5.

4.1 Proximity neighbour selection

PNS aims at keeping the contacts with the least cost in the routing table. As the capacity of each bucket is limited to k contacts, a new contact with low cost that is to be added to a full bucket replaces a contact with higher cost.

The contacts saved in the buckets of a Plain Old Kademlia (POK) routing table are essentially random. The only requirement is that the contacts' overlay identifiers be consistent with the prefix of the respective buckets. Applying a cost function to the contacts of a bucket corresponds to making a choice between contacts that are considered equivalent by the routing table, and thus does not alter the correctness of the bucket's entries.

Each Kademlia peer continuously learns of new peers from incoming requests as well as during own (iterative) lookups. For this reason, we do not implement additional algorithms to actively search for better peers, but rather opt for a passive strategy that simply retains the best peers seen so far by Kademlia's built-in mechanisms. Consequently, this PNS algorithm does not cause additional traffic.

Given this replacement strategy, an attacker that manages to forge its cost value has higher chances of intruding into the routing tables of other peers. Underlay metrics that rely on data supplied by the sender of a message are vulnerable to this kind of attack. However, if the receiver calculates the cost value with its own data, attacks can only come from close peers, as peers from distant locations are penalised by the underlay metric. This effectively results in better protection than in POK, which does not distinguish the underlay location of peers.

4.2 Proximity route selection

The objective of PRS is to route queries to peers with low cost. Such an objective results in a trade-off between the overlay distance between peers, which is measured in the XOR metric, and the underlay distance, which is measured by an underlay metric. While the overlay distance expresses the distance to the information that is being looked up, the underlay distance gives an estimation for the cost of communication.

As routing in Kademlia is iterative, the initiator of a lookup determines each hop. It chooses the contact with the least cost from a set of contacts that it considers equivalent with respect to the overlay metric. For simplicity, we define to be equivalent the k currently known closest peers to the lookup destination. Thus, in step 2 of the lookup algorithm from Section 3.2, the lookup initiator chooses not the closest contact, but rather, the contact with the least cost as the next hop.

We will now analyse the correctness and complexity of this approach. First, we note that the set of the k clos-

est peers can be seen as a window that moves toward the lookup destination. Both POK and PRS select from this set the next peer that will be queried. While POK always queries the peer that is closest (in the XOR metric) to the lookup destination, PRS queries the peer with the least cost. The reply to a request roughly corresponds to the contents of a queried peer's bucket. As we have seen above, the contacts contained in a bucket are equivalent in that their identifiers all share the prefix of the bucket, which is at least one bit longer than the prefix of the previous hop. Close to the lookup destination, some buckets of the queried peers might not be full, given that fewer than k peers exist with the prefix of the bucket. In order to return k contacts, a queried peer will then select peers from other buckets as well, leading to some peers not being one more prefix bit closer than the previous ones. However, this effect occurs independent of n, the number of peers. It follows that the PRS routing algorithm has the same lookup complexity as POK, that is, $O(\log n)$ hops.

Kademlia benefits from so-called *random bit improvement* [21]. This refers to the probability of a bucket containing a contact that improves the common prefix to the lookup destination by more than the one bit guaranteed by the prefix of the bucket. For instance, Stutzbach et al. [21] report an average improvement of 5.7 bits per hop with $k = 20$. Our heuristic of always selecting the contact with the least cost limits this effect and leads to more messages being sent[1]. This can be tolerated, however, as long as sending many low-cost messages does not exceed the cost of sending a single message that covers a larger XOR distance.

5 Underlay Metrics

An underlay metric provides information about the underlay network to the algorithms of the overlay network. Having described proximity algorithms that use the abstract underlay metric interface in Section 4, we will now turn to implementations of this interface.

The goal of an underlay metric is to optimise routing according to the needs of a specific application. For instance, routing could be optimised to maximise the locality of traffic, reduce the latencies of lookups, prefer communication with peers that have high-bandwidth connections, or avoid routing to untrustworthy peers.

To achieve one or more of these goals, an underlay metric has several possibilities to base its calculations on.

- First, it can use data gained during its *own measurements*. However, as routing in Kademlia is iterative, the cost value has to be calculated without noticeable time delay. Thus, when a cost value is requested for a previously unknown peer, the underlay metric implementation cannot send extra probe messages in order to determine, for instance, the round-trip time. Instead, it may return a default cost value, if appropriate. Otherwise, another type of implementation has to be chosen.

- The second possibility is to make use of *data provided by correspondent peers*. For instance, network coordinates as used by Vivaldi [4] (to which we will come back later in this section) can be appended to the contact information that is already contained in the messages exchanged by POK. External means are possibly necessary to secure this approach against the attacks detailed in Section 4.1.

- Finally, a peer can use a *local database* to look up information about other peers. For instance, such a database could contain a mapping from IP address ranges to ISP networks.

Table 1. Calculation of the Cost Value in the Cluster and Vivaldi Underlay Metrics.

Underlay Metric	Peers located in same...	Cost
Cluster	ISP	0
	Region	1
	Country	2
	Continent	3
	World	4
Vivaldi	World	RTT

In the following, we introduce a novel Cluster metric that looks up geographic information about peers in a local database to achieve locality of traffic. We furthermore give an overview of Vivaldi, which predicts round-trip times based on network coordinates. Table 1 summarises the way both underlay metrics calculate cost values.

Vivaldi has been devised to reduce lookup latencies, which is not the primary optimisation goal of this paper. We discuss Vivaldi, nevertheless, to demonstrate how it can be used as an underlay metric even though it requires exchange of network coordinates. Furthermore, we will show in Section 7 that low latencies do not imply locality of traffic.

5.1 Cluster

In order to provide for locality of traffic, the Cluster underlay metric needs to obtain geographic information about peers. With GeoIP [10], MaxMind offers a downloadable

[1] By giving full priority to the underlay metric, the bit improvement of a hop is now bounded below by the minimum bit improvement of all k closest contacts. In contrast, POK always achieves the maximum available bit improvement.

database that associates IP addresses with geographic information such as the latitude and longitude, ISP, region, and country.

When a peer needs to calculate the cost for routing to a given peer, it looks up the distant peer's IP address in a local copy of the database. If the local and distant peers share the same ISP, we assign the least cost ($c = 0$). The cost is increased by one if both are not located in the same ISP, but only the same region ($c = 1$). The cost for peers located in the same country ($c = 2$), the same continent ($c = 3$), or for peers having nothing in common at all ($c = 4$) is calculated analogously.

The primary intention of the Cluster underlay metric is to keep the largest possible fraction of messages inside the subnet of the peer's own ISP, and send the least possible amount of messages to peers on other continents. However, it is often reasonable to assume, for instance, that the latencies between peers within the same ISP are lower than the latencies between peers that are just located in the same country. Thus, by applying the Cluster underlay metric, in addition to ensuring locality of traffic, one is likely to reduce lookup latencies as well.

5.2 Vivaldi

In Vivaldi [4], each peer maintains a network coordinate. A peer's network coordinate locates it in a multidimensional round-trip time space. Initially, all network coordinates are random. Each time a peer measures the round-trip time to another peer, it updates its own coordinate with knowledge of the measured round-trip time and the coordinate of the distant peer. In our implementation, round-trip times (including processing time at the distant peer) are observed when regular lookup messages are sent and the corresponding replies are received by the initiator of a lookup.

Knowledge of a distant peer's network coordinate allows a peer to estimate the round-trip time to the distant peer. We embed network coordinates into regular messages, alongside information that is already exchanged in POK, e.g., the overlay identifier and the IP address. Thus, each time a peer learns of another peer, it automatically receives its network coordinate as well.

When a peer needs to determine the next hop in a lookup, it already knows the network coordinates of all candidate peers. Their round-trip times can now be estimated without time delay and are used by the Vivaldi underlay metric as cost values. If regular lookup messages are sent sufficiently often, no additional messages need to be exchanged for round-trip time measurements. However, the size of messages increases to carry the network coordinates of the peers mentioned in the message.

6 Methodology

Having discussed proximity algorithms and two exemplary underlay metrics, we want to evaluate the extent to which they permit the achievement of locality of traffic. This section details the design of our experiments.

While an evaluation could be carried out on a distributed test-bed or by modifying the client software of a deployed P2P network, these approaches have strong limitations in our context, and so we resort, instead, to simulation. The limitations are exemplified by the PlanetLab [14] testbed, which consists of about 800 nodes—too few for the analysis of a P2P network [18]. Furthermore, the geographic distribution of PlanetLab's nodes does not appear to be representative for currently deployed P2P networks. Similarly, measurements in the Azureus or Kad networks, for instance, are only feasible with locally modified client software. Due to Kademlia's iterative routing, the PRS routing algorithm can easily be implemented in a measurement client. In contrast, PNS requires that the routing tables of all participants be modified so that they are effective on the contact lists returned by queried peers. This means that we would not be able to use PNS when evaluating in a deployed DHT. We suspect, however, that the use of PRS is sensible only in conjunction with PNS.

For all these reasons, we implemented Kademlia and our extensions in PeerfactSim [13], a large-scale P2P simulator developed at Technische Universität Darmstadt. Section 6.1 gives an overview of its novel underlay model, which integrates a large amount of measurement data in order to realistically model the delay, jitter and packet loss probability between hosts. Section 6.2 describes the simulated scenario and contains further implementation details.

6.1 Underlay Model

In order to reflect the network characteristics of the Internet, we integrated measurement data from two Internet measurement projects, CAIDA [3] and PingER [22], into our simulation framework PeerfactSim.

The data taken from the CAIDA Macroscopic Topology Project consists of 40 GB of round-trip time measurements from 20 monitor hosts distributed all over the word, to about 400,000 destination hosts. In order to allow for an efficient simulation, we applied Global Network Positioning [12] to place the monitor and destination hosts into a multidimensional Euclidean space. With each host being characterised by a point in this space, the delay between any two hosts can be predicted using an adequate distance function.

The MaxMind GeoIP database allows us to look up the IP addresses of the destination hosts and find out their geographic position, i.e., continent, country, region, and ISP. The location data of peers can be used to set up simulation

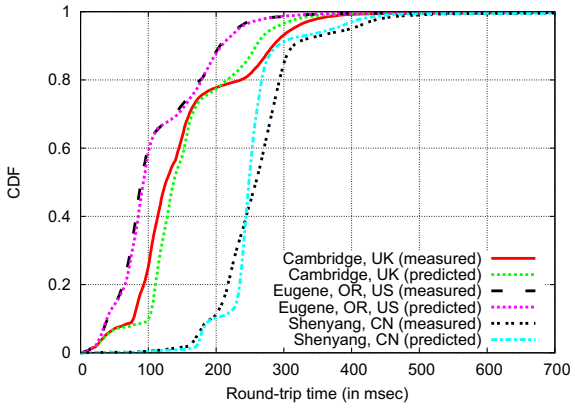

Figure 1. The measured and predicted round-trip time distribution as seen from three different locations in the world.

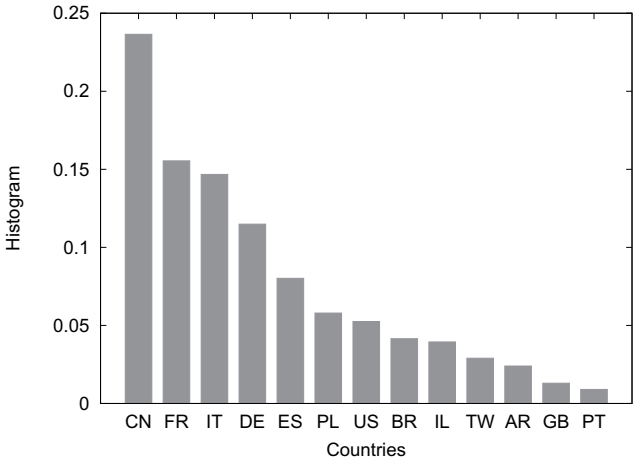

Figure 2. Histogram of the geographic distribution of the peers used in our experiments.

scenarios with a precise geographic distribution of peers.

We furthermore use the measurement data of the IEPM PingER Project to fix packet loss probabilities and inter-packet delay variations (jitter) depending on the geographic location of the sender and receiver.

As an illustration of our results, Figure 1 depicts the round-trip times predicted by our underlay model and the measured round-trip time distribution for the Internet as seen from three CAIDA monitors that have not been used in constructing our underlay model. We observe that the predicted round-trip time distribution accurately matches the distribution measured by the monitor hosts. Moreover, it can be seen that the round-trip time distribution varies substantially and qualitatively in different locations in the world. We believe that this characteristic is not accurately captured by network topology generators.

6.2 Experimental Design

The goal of our evaluation is to determine the extent to which locality of traffic can be improved by making use of the Cluster and Vivaldi underlay metrics. To this end, we compare POK to the underlay metric-enabled variants that make use of PNS and a PNS+PRS combination.

In order to evaluate the underlay metrics in a scenario that is close to reality, we find it is essential to consider a realistic geographic distribution of peers. In a measurement study of the Kad network, Steiner et al. [19, 20] observed a peer distribution from which we derived our scenario[2] (Figure 2). Additionally, we model peer turnover according to a Weibull distribution that has been observed in the same study. The protocol parameters of Kademlia are set to $k = 20$, $b = 2$, and $\alpha = 3$. Finally, we disable caching of data items on peers other than the k peers that are closest (in the overlay metric) to the lookup destination.

[2]In contrast to [20], our simulation does not include peers from South Korea as no data was available in our underlay model. The percentage of missing peers has been allocated to the remaining countries.

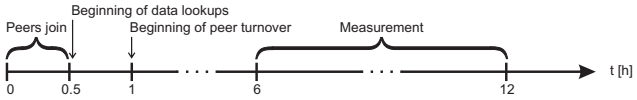

Figure 3. The course of the simulation.

The sequence of the simulation scenario is depicted in Figure 3. We set up the Kademlia network with 10,000 peers during the first hour. Each peer initially obtains a set of 100 entirely random contacts to fill its routing table. It then starts to look up data items in regular intervals of 10 minutes. The keys to be looked up are selected following a Zipf distribution from a pool of 10,000 data items. With peer turnover enabled, the simulation runs for five hours, after which we take six hours to carry out our measurements.

7 Evaluation

To summarise the results of our simulation, we first note that all variants of Kademlia achieved lookup success rates of around 99.9 %. In order to render the measurement results about the respective underlay metrics comparable, we tuned the length of message timeouts in the simulation so that on average, each underlay metric causes the same cost in terms of messages sent per lookup. As POK chooses hops optimally with respect to the overlay metric, the number of messages is generally lower than what can be achieved with underlay metrics enabled (by one third in our simulation). However, in the following, we will show that the

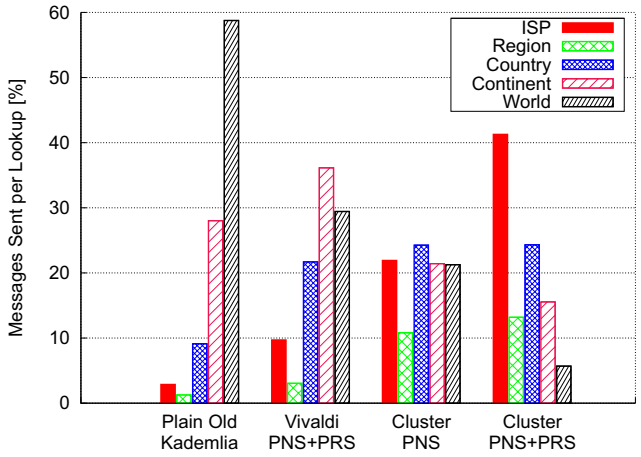

Figure 4. The distribution of messages sent during data lookups.

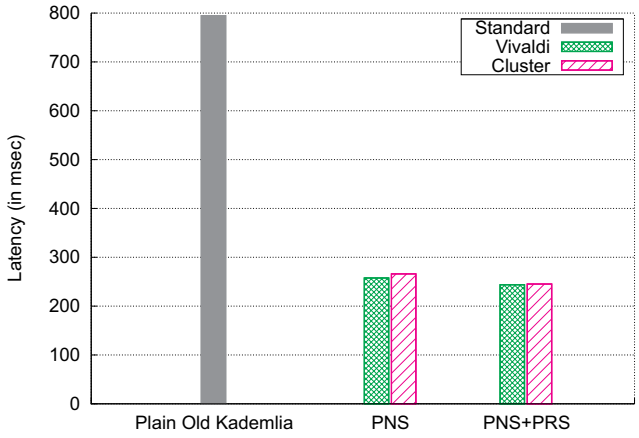

Figure 5. The average latencies of successful data lookups.

geographic distribution of message receivers is far from optimal in POK.

Figure 4 plots the geographic distribution of the messages sent during data lookups. Almost 60 % of the traffic caused by POK is intercontinental, and less than 15 % of all messages are sent to peers in the same country. The fraction of messages that are exchanged within the peer's ISP is less than 3 %.

As Vivaldi optimises routing for latency reduction, one might be inclined to guess that it will find many low-latency peers in the same ISP. However, our analysis shows that this is not true. The number of messages sent to intercontinental peers by Vivaldi is only half as many as those sent by POK, but the highest growth can be observed in the messages exchanged between peers in the same country and on the same continent. Although this already considerably reduces the load in intercontinental backbones, still more than 90 % of all messages leave the ISP's subnet.

In this regard, the Cluster metric with only PNS performs better than Vivaldi. This is evident from the fact that in the former, more than 20 % of all messages do not leave the ISP's network and the fraction of intercontinental and continental messages is lower. Yet the Cluster metric obtains its best result with a combination of PNS and PRS. With PNS+PRS, more than 40 % of all messages remain in the ISP's network, and less than 6 % are sent through intercontinental backbones.

The geographic message distributions of both Vivaldi and the Cluster underlay metric display the same behaviour with regard to PNS and PNS+PRS (although not shown for Vivaldi): While PNS already results in an improvement, the performance of both can be further ameliorated by additionally making use of PRS.

In contrast, this result is not valid for the latencies of successful data lookups as shown in Figure 5. While PNS reduces the lookup latencies of POK by 66 % with either underlay metric, the improvement from PNS to PNS+PRS is only marginal. Similarly, the performance of the underlay metrics themselves does not differ significantly.

In the absence of figures for the DHT fraction of the traffic caused by currently deployed P2P networks, we cannot predict how much ISPs would benefit from reduced cross-network DHT traffic if existing applications used location-aware routing. In general, this largely depends on how an application makes use of the DHT. In (future) applications that require frequent lookups, the traffic caused by the DHT might be considerable in contrast to applications that make use of the DHT as a fallback system only.

As to the Cluster underlay metric, several issues are open for improvement. First, the ISP data of MaxMind's GeoIP is a commercial offering. In order to enable deployment in an open-source P2P network, this data could be collected by the community. For instance, each user could configure the P2P software by indicating his own ISP. However, this comes at the cost of the security issues discussed in Section 4.1. Second, the size of ISPs, regions, and countries is not equal. For instance, there are regions in the US that are larger than all of Germany. We feel that a community-based approach offers the opportunity of collecting more precise data that could even further improve locality.

8 Conclusion & Future Work

This work applied proximity neighbour selection (PNS) and proximity route selection (PRS) to iterative routing. We devised PNS and PRS algorithms for Kademlia and showed

that they maintain Kademlia's structural properties. The algorithms do not require a special selection of bootstrapping contacts and work with a passive neighbour discovery strategy.

An underlay metric parameterises the PNS and PRS algorithms to optimise routing toward a specific goal. We considered locality of traffic and introduced a novel Cluster underlay metric that prioritises peers from the same ISP.

An evaluation based on Internet measurement data and a scenario derived from observations of the Kad network contrasted the performance of the Cluster underlay metric and Vivaldi. We found PNS+PRS to offer no significant advantage in terms of the lookup latency, but to massively improve the locality of traffic when compared to PNS alone. While Vivaldi and the Cluster metric reduce the lookup latency of Kademlia to the same extent, the Cluster metric outperforms Vivaldi when it comes to locality of traffic. We were able to show that low latencies do not imply locality of traffic. Furthermore, we showed that locality of traffic can be implemented without negatively impacting the lookup latencies.

We conclude that embracing the peer next door is indeed a promising approach. Users of the P2P application benefit from lookup latencies that have been divided by three, and Internet service providers can reduce their costs because 40 % of the lookup traffic does not leave their domain.

The use of the PRS routing algorithm for Kademlia is not limited to locality. Depending on the needs of the application, other underlay metrics, or a combination of them, can be used to optimise routing toward other goals. For instance, the underlay metrics could strive for "trusted routing" or select peers with high-bandwidth connections preferentially.

We are currently enhancing the routing algorithm to incorporate a tunable trade-off between the underlay and overlay metrics, the trade-off being the number of messages incurred by neglecting the XOR metric versus the optimisation of routing achieved by applying an underlay metric.

Acknowledgements

We would like to thank Gerald Klunker for his valuable work on the underlay model.

References

[1] M. Castro, P. Druschel, Y. C. Hu, and A. Rowstron. Exploiting network proximity in distributed hash tables. Technical report, Microsoft Research, 2002.

[2] D. R. Choffnes and F. E. Bustamante. Taming the torrent. In *SIGCOMM '08*, 2008.

[3] Cooperative Association for Internet Data Analysis (CAIDA). Macroscopic Topology Project. `http://www.caida.org/analysis/topology/macroscopic/`.

[4] F. Dabek, R. Cox, F. Kaashoek, and R. Morris. Vivaldi: a decentralized network coordinate system. In *SIGCOMM '04*, 2004.

[5] J. Falkner, M. Piatek, J. P. John, A. Krishnamurthy, and T. Anderson. Profiling a million user DHT. In *IMC '07*, 2007.

[6] K. Hildrum, J. D. Kubiatowicz, S. Rao, and B. Y. Zhao. Distribted object location in a dynamic network. In *SPAA '02*, 2002.

[7] S. Jain, R. Mahajan, and D. Wetherall. A study of the performance potential of DHT-based overlays. In *USENIX '03*, 2003.

[8] T. Karagiannis, A. Broido, N. Brownlee, K. C. Claffy, and M. Faloutsos. Is P2P dying or just hiding? In *GLOBECOM '04*, 2004.

[9] Light Reading, Controlling P2P Traffic. `http://www.lightreading.com`.

[10] MaxMind Geolocation Technology. `http://www.maxmind.com/`.

[11] P. Maymounkov and D. Mazières. Kademlia: A peer-to-peer information system based on the XOR metric. In *IPTPS '02*, 2002.

[12] T. Ng and H. Zhang. Towards Global Network Positioning. In *IMW '01*, 2001.

[13] PeerfactSim: A Simulator for Large-Scale Peer-to-Peer Networks. `http://www.peerfactsim.com/`.

[14] PlanetLab. An Open Platform for Developing, Deploying, and Accessing Planetary-Scale Services. `http://www.planetlab.com`.

[15] Plaxton, Rajaraman, and Richa. Accessing nearby copies of replicated objects in a distributed environment. *MST: Mathematical Systems Theory*, 1999.

[16] S. Ratnasamy, M. Handley, R. M. Karp, and S. Shenker. Topologically-aware overlay construction and server selection. In *INFOCOM '02*, 2002.

[17] S. Rhea, D. Geels, T. Roscoe, and J. Kubiatowicz. Handling churn in a DHT. In *ATEC '04*, 2004.

[18] N. Spring, L. Peterson, A. Bavier, and V. Pai. Using planetlab for network research: myths, realities, and best practices. *ACM SIGOPS Operating Systems Review*, 2006.

[19] M. Steiner, T. En Najjary, and E. W. Biersack. Analyzing peer behavior in KAD. Technical report, Institut Eurecom, 2007.

[20] M. Steiner, T. En-Najjary, and E. W. Biersack. A global view of KAD. In *IMC '07*, 2007.

[21] D. Stutzbach and R. Rejaie. Improving lookup performance over a widely-deployed DHT. In *INFOCOM '06*, 2006.

[22] The PingER Project. `http://www-iepm.slac.stanford.edu/pinger/`.

[23] H. Xie, Richard, A. Krishnamurthy, Y. Liu, and A. Silberschatz. P4P: Provider portal for applications. In *SIGCOMM '08*, 2008.

[24] B. Y. Zhao, A. D. Joseph, and J. Kubiatowicz. Locality aware mechanisms for large-scale networks. In *FuDiCo '02*, 2002.