Work in Progress: Where is my Peer? Evaluation of the Vivaldi Network Coordinate System in Azureus

Moritz Steiner[1]* and Ernst W. Biersack[2]

[1] Bell-Labs / Alcatel Lucent
791 Holmdel-Keyport Rd
Holmdel, NJ 07733, USA
moritzs@alcatel-lucent.com
[2] Eurecom
2229, route des Crêtes
06904 Sophia–Antipolis, France
erbi@eurecom.fr

Abstract. Network coordinates allow to estimate the latency among a large number of hosts in a scalable way. Recently, Azureus, a popular implementation of BitTorrent, has implemented network coordinates. We have developed a crawler that allows us to obtain from the network coordinates over one hundred thousand peers running Azureus and to measure the network and application level round trip times to these peers.
Our measurements confirm that network coordinates allow to correctly estimate the round trip time between two peers. Our measurements also show that the round trip times from our crawling host to a set of peers located in the same country can vary between a few tens of milliseconds to more than one second. This high variance is due to the large buffers in the ADSL access links, which can increase the round trip time by hundreds of milliseconds. As a consequence, network coordinates and round trip estimations in general cannot be used to select peers that are "nearby", such as peers connected to the same ISP or located in the same country.

Keywords: peer-to-peer, measurement, network coordinate system

1 Introduction

Internet Coordinate Systems [1–4] are very popular today, since selecting nodes based on their location in the network is a basic building block for many distributed systems. The Euclidean distance between the coordinates of two hosts is used as an estimator of the round trip time between these hosts; the actual measurement needs not to be done. In this paper, we consider Azureus whose peers use the network coordinate system Vivaldi [1] to compute their network

* This work was done while the first author was working towards his Ph.D. at Eurecom

coordinates. With the help of these coordinates, they try to find other peers physically close to them in order to reduce download times and to keep the traffic local, e.g. inside an ISP or a country.

Our study aims at evaluating the Vivaldi coordinate system in "action". By crawling Azureus, we collect round trip times at the application layer and at the network layer of several hundred thousand of clients around the world. We also collect their Vivaldi coordinates and analyze their accuracy. Previously, only the work of Ledlie et al. [5] evaluated the Vivaldi coordinate system in the "wild" and introduced what is known as the *second version* of the Vivaldi coordinate system. However, all the measurements in [5] are performed from PlanetLab nodes. Since most users are connected to the Internet with ADSL, we feel that these results are biased (cf. Sect. 4.3).

As we will discuss in detail, our results indicate that the Vivaldi network coordinates of Azureus are well suited to predict the round trip times between two Azureus hosts. However, Vivaldi network coordinates often do not allow to infer the geographical distance between hosts. Extremely long round trip times of several seconds are a strong indication for heavily loaded ADSL links and not for a large distance between those peers. Due to this fact, groups of peers in geographical proximity, e.g. countries or ISPs, are not reflected in the coordinate space and it is not possible to find close by peers based on their Vivaldi coordinates in Azureus.

In Sect. 2, we give a brief background on the use of Kademlia [6] based peer-to-peer systems and on the Vivaldi network coordinates system. In Sect. 3, we detail our measurement methodology to learn about the peers present in the Azureus network, to get their Vivaldi coordinates, and to measure the application layer and the network layer round trip times to those peers. In Sect. 4, we examine how well the application layer round trip times used by Azureus to compute the Vivaldi coordinates are correlated to the network layer round trip times. Moreover we analyze the accuracy of the network coordinates. Finally, in Sect. 5, we conclude and recall an alternative approach to find close by peers.

2 Background

2.1 Azureus

Distributed Hash Tables (DHTs) map a large identifier space onto the nodes that participate in the system in a deterministic and distributed way. The DHT Kademlia [6] is implemented by several peer-to-peer applications such as Azureus [7], Overnet [8], eMule [9], aMule [10], and lately the Storm worm [11]. The two open–source projects eMule and aMule share the same implementation of Kademlia and they do have the largest number of simultaneously connected users, 3 – 4.5 million users [12], followed by Azureus with about 1 million users.

2.2 Vivaldi Network Coordinates

Internet coordinate systems allow a host to predict the round trip times to other hosts without actually measuring them. Explicit measurements are of-

ten unattractive, because the cost of measurement can outweigh the benefits of exploiting proximity information. Coordinates are assigned to hosts such that the distance between their coordinates predicts the RTT between these hosts. Simulation–based systems map nodes and latencies into a physical system whose minimum energy state determines the node coordinates. Vivaldi [1] calculates the coordinates as the solution to a spring relaxation problem.

The system envisions a spring between each pair of nodes with the resting position of the spring equaling the network latency between the pair. Each node updates and refines its own position successively by taking into account newly reported RTT measurement toward its communication partners. Since this information is piggybacked on other network messages, e.g. route requests, no additional messages are sent through the network. In other words, nodes allow themselves to be pulled or pushed by a connected spring. Vivaldi attempts to minimize the potential energies over all springs.

Vivaldi uses Euclidean coordinates of d dimensions augmented with a height value: $x = x_1, \ldots, x_d, x_h$. The coordinates without the height vector can be seen as reflecting the distance across the high–speed Internet core to which the end users are attached. The last mile that may suffer from queuing delays due to large buffers, as it is the case for ADSL, is represented by the height value. Without using the height, the coordinate space would be erroneous since it is possible to measure latencies in the order of seconds between peers in the same country, which is more than the propagation time needed to make the tour of the globe (≈500 ms). To calculate the distance between two nodes x and y, first the distance of their Euclidean coordinates is calculated and then the heights of both nodes are added:

$$Vivaldi\ distance(x, y) = \sqrt{\sum_{i=1}^{d} (x_i - y_i)^2} + x_h + y_h$$

For further details on Vivaldi we refer the reader to [1].

3 Measurement Methodology

3.1 Crawlers

The first step in order to do any latency measurements is to learn about the hosts we want to measure to. Therefore, we used our crawler Blizzard for KAD [12] and adapted it to work for Azureus as well. For each peer P, our crawler logs the time of the crawl, the IP address of P, the port number used for the control traffic, the DHT ID of P, and the Vivaldi network coordinates of P.

The implementation of Blizzard is straightforward: it runs on a local machine and starts by contacting a seed which is run by us. The crawler asks the seed peer for a set of peers to start with and uses a simple breadth first search and iterative queries. It queries the peers it already knows in order to discover new peers.

At the beginning of each crawl, the number of newly discovered peers grows exponentially before it asymptotically approaches the total number of peers in the system. The crawl is done when no new peers are discovered and all the discovered peers have been contacted.

The crawler implements two asynchronous threads: one thread sends the `REQUEST_ FIND_NODE(id)` messages and the other thread receives and parses the `REPLY_FIND_NODE` messages returned. A list containing all the peers discovered so far is used and maintained by both threads: the receiving thread adds the peers extracted from the `REPLY_FIND_NODE` messages to the list, whereas the sending thread iterates over the list and sends n (8 to 16) `REQUEST_FIND_NODE(id)` messages to every peer in the list. The value of the DHT ID `id` is different in each message. This is done in order to minimize the overlap between the sets of peers returned.

Not all the peers discovered can be contacted directly by the crawler. Approximately half of the peers queried do not respond to the crawler. There are two main reasons why a peer does not respond to our queries: either the peer has left the system or the peer is behind a NAT that blocks our query. For the crawler, it is not possible to distinguish between these two cases.

To crawl the DHT implemented by a BitTorrent client is a new approach to learn about BitTorrent peers, since to this day the classical approach was to learn about torrents on portal web sites and to contact the corresponding trackers afterwards. Since in BitTorrent every torrent forms a swarm of peers, all existing torrents need to be tracked in order to get a full view on all clients participating. Our approach is not based on torrents, but we make use of the DHT in which all peers participate and in which information about all torrents is published. For the moment, this DHT is used as a fall-back mechanism in case the tracker goes down. Not all DHTs implemented by the different BitTorrent clients are compatible with each other. Therefore, with our method of crawling a DHT, we do not learn about all peers using BitTorrent, but only about those using a DHT compatible to the DHT of Azureus.

3.2 Application Layer Round Trip Time

The crawler exploits the control messages of the DHT and uses the messages intended for routing in order to learn about other peers. The *application layer round trip time* (ARTT) of those messages is composed by two parts: the network layer round trip time (NRTT), which will be explained in the next paragraph, and the additional delay induced by the computation of the application that depends on the load of the end-system (Figure 1). Therefore, we always expect the application layer round trip time to be larger than the network layer round trip time. In the remaining of this chapter, the round trip time is always expressed in milliseconds.

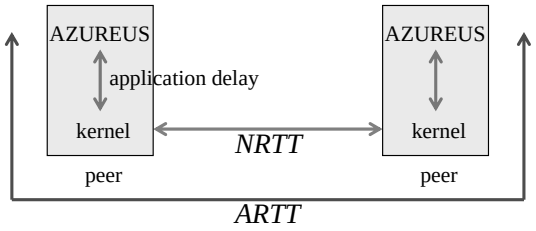


Fig. 1. The ARTT is composed of the NRTT and the additional delay induced by the application.

3.3 Network Layer Round Trip Time

The `REPLY_FIND_NODE` message contains the IP address and the port of a peer. After the discovery of a peer, a TCP packet with the ACK flag set is sent on this very port. The peer is expected to reply with another TCP packet having the RST flag set [13]. We call the delay between the emission of the TCP ACK and the reception of the TCP RST the *network layer round trip time* (NRTT). The TCP ACK packet we send is directly processed by the kernel of the operating system of the queried peer. Therefore, we expect the network layer round trip time to be equal to the round trip time of an ICMP ping. We send a TCP ACK instead of an ICMP ping since some network providers filter ICMP packets and most (Wlan-) routers and personal firewalls do not reply to ICMP pings. Moreover, by default, all ports are closed, which is why it is so important to learn about an open port by first crawling the peer-to-peer network.

3.4 Network Coordinates

The reply messages received during a crawl also contain the Vivaldi [1] network coordinates of the queried peer. Depending on the version of the Azureus client, different versions of the Vivaldi network coordinates are communicated: none, version 1 (made of 2 dimensions plus the height), or version 1 and version 2.4 (made of 4 dimensions plus the height). The difference between the two implementations of the Vivaldi network coordinate system is not limited to the number of dimensions, also the age of the coordinate is transmitted. Moreover, the ways new measurements are used to update the coordinates have become more sophisticated. In total, 16 additional bytes are transmitted for version 2.4. See [5] for details about the different versions of the network coordinates. 97.2% of the Azureus clients use the latest version of the DHT protocol which transmits both version 1 and version 2.4 of the network coordinates. Therefore, we considered only these peers.

We run an Azureus client on the machine performing the crawl in order to learn about the network coordinates of the crawler itself. Using those coordinates, we are able to compute the Vivaldi distance between the crawler and the queried

peers. It is expected to approximate the round trip time of the packets sent to this peer. Since the Azureus application is not aware of the NRTT, but only of the ARTT, we expect the Vivaldi distance to be more tightly correlated with the ARTT than with the NRTT.

4 Results

For the analysis presented in this paper, we collected the following datasets.

- **Mannheim:** The crawler is at the University of Mannheim, which is attached to the German research network. One single crawl performed on March 31, 2008, at 08:00 CET. 1,044,155 peers have been discovered, 291,850 of them responded to the crawler (Azureus ARTT and network coordinates are available). 157,205 peers replied to the TCP ACK. For those peers, the full data is available. The crawl duration was 12 minutes. This dataset is available online:
  `http://www.eurecom.fr/~btroup/vivaldiazureus/`.
- **Eurécom:** The crawler is at Eurécom, which is attached to the French research network. Starting on 15th of February, 2008, we performed 3 full crawls of the Azureus network a day (05:00, 13:00, and 21:00 CET). On each crawl, 1 – 1.4 million were discovered. About 300,000 – 400,000 of them responded to the crawler, thus, of those peers, Azureus ARTT and network coordinates are available. For about 50% of the responding peers, the NRTT is also available, the other peers did not respond to the TCP ACK packet sent. Each crawl has a duration of about 20 minutes.
- **ADSL:** The crawler is connected via an ADSL line. One single crawl was performed on March 26, 2008. This crawl took 12 hours and, out of 1,267,822 discovered peers, 118,548 peers responded. Whereas the NRTTs are only available for 37,346 of those peers.

Table 1 gives an overview of the results obtained on Azureus in the Mannheim dataset. The results of the two other vantage points are omitted for space constraints since they are qualitatively very similar. We mapped the IP addresses of the peers to their countries using Maxmind [14], a database containing geolocation information. In the table, we list several countries from different parts of the world that are representative for their continents: countries close to our crawling site, countries far away, countries with good and with poor Internet connectivity. The second column shows the number of unique Azureus clients measured, the third column shows the average ARTT followed by 5 columns for the NRTT. Columns 9 and 10 show the average Vivaldi distance from our crawling site to the Azureus peers.

The mean NRTT value does not allow to distinguish between continents, compare 566 ms for Spain with 319 ms for the US or 377 ms for Korea. In fact, the *mean* NRTT is strongly biased by outliers, as can be seen from the 95th percentile (Column 8 and Fig. 2). A much better indicator for geographic proximity is the 5th percentile: 21 to 57 ms for European countries, 111 to 120 ms for North

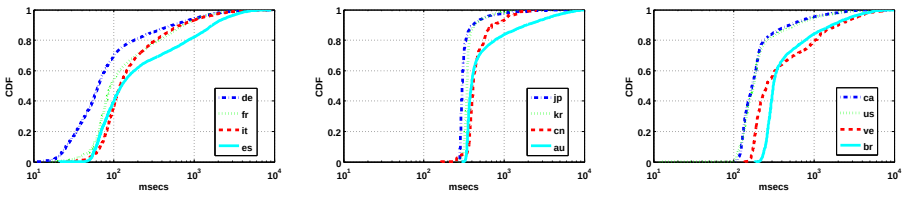
Table 1. Overview of the Azureus results: ARTTs, NRTTs, and coordinate distances in milliseconds. Measurement host is located in Mannheim.

1	2	3	4	5	6	7	8	9	10
Country	# Clients	ARTT	NRTT					Coordinates	
		mean	mean	st. dev.	5th perc.	median	95th perc.	v2	v1
France	13,775	359	306	626	*44*	92	1,235	542	344
Germany	11,439	435	236	598	*21*	64	1,143	736	415
Spain	8,281	641	566	984	*55*	125	2,604	1,043	581
Italy	3,464	389	325	671	*57*	119	1,286	560	368
Canada	12,349	360	298	512	120	169	948	454	259
US	32,528	394	319	543	111	176	1,052	488	269
Venezuela	530	851	765	1,175	169	258	3,299	1,197	657
Brazil	2,364	776	718	1,067	233	312	2,828	1,053	598
China	315	563	513	302	324	413	1,101	656	381
Korea	362	413	377	134	308	346	*563*	372	231
Japan	1,283	443	370	358	281	300	*586*	466	246
Australia	3,733	934	872	1,326	340	392	3,729	1,162	634
All countries	157,205	454	375	739	37	151	1,541	647	376

America, 169 to 233 ms for South America, and 281 to 340 ms for Asia and Australia. Even if it is possible to make a difference between the NRTT distributions for different continents, this does not imply that, from the NRTT to a single peer, one can deduce its continent of origin.

We do explain the very low variance for peers in Korea with the widely deployed fiber to the home in these countries. The high variance of the NRTTs, e.g. in Europe, is introduced by the last mile to these users that are often connected with ADSL [15]. The ADSL access links own buffers that can add an additional delay ranging from tens of milliseconds to more than a second.

Compared to the mean NRTTs (Column 4), the mean ARTTs (Column 3) are slightly higher and they show a higher variance, which reflects the additional delay introduced by the Azureus application. However, the overall shape of the corresponding cumulative distribution function per country remains the same. In Fig. 2, the measured CDF of the NRTTs are plotted.



(a) Europe (b) Asia (c) America

Fig. 2. NRTT for different countries. Origin of the measurements is in Mannheim / Germany.

4.1 Network Coordinates

We compared the calculated Vivaldi distances for both implementations of the coordinate system to the application round trip time measurements and to the network layer round trip time measurements we performed.

Table 2. Correlations of the results for RTTs and Vivaldi distances shown in table 1.

1	2	3	4	5	6	7
	ARTT NRTT	v.1 v.2	v.2 ARTT	v.2 NRTT	v.1 ARTT	v.1 NRTT
France	0.94	0.89	0.91	0.89	0.85	0.84
Germany	*0.65*	0.91	0.92	0.63	0.87	0.61
Spain	0.93	0.94	0.94	0.89	0.91	0.87
Italy	0.90	0.92	0.91	0.89	0.85	0.84
Canada	0.85	0.81	0.83	0.81	0.77	0.74
US	0.81	0.86	0.85	0.82	0.79	0.74
Venezuela	0.97	0.84	0.93	0.94	0.79	0.80
Brazil	0.94	0.90	0.92	0.93	0.87	0.88
China	0.75	0.80	0.93	*0.54*	0.77	0.63
Korea	0.94	0.74	0.83	0.80	0.60	0.59
Japan	*0.48*	0.90	0.93	0.47	0.93	0.47
Australia	0.98	0.87	0.91	0.91	0.85	0.84
All countries	0.88	0.89	0.90	0.84	0.84	0.80

The ARTTs and the NRTTs do have a positive linear correlation of 0.88 (Table 2, column 2). For some countries, such as Germany and Japan, the correlation between the ARTTs and the NRTTs is much lower than compared to other countries. Therefore, the correlation between the ARTTs and the distances computed with the coordinates are lower, too. The weak correlation for the German peers is not due to the measurement origin being in Germany, the other two datasets also confirmed these results.

The correlation between the two versions of the network coordinates is 0.89 (Column 3). This strong correlation indicates that the two additional dimensions introduced in version 2 do not have a big impact. The correlation between the coordinates version 1 and the ARTTs is 0.84 (Column 6), for version 2 this increases to 0.90 (Column 4). The correlation between the network coordinates version 1 and the NRTT is 0.80 (Column 7), for version 2 this value increases to 0.84 (Column 5). We can conclude that the two additional dimensions, the introduced age of the coordinates, and the resulting additional overhead in version 2 do not result in a significant improvement of the network coordinates' accuracy.

Our direct measurements of the NRTT are all taken from hosts based in Europe, thus the CDFs for the different countries do all have Europe as a point of origin (Figure 2). To get a different point of origin, we need to make use of the *network coordinates*. We chose an Azureus client in the US and computed its Vivaldi distance to all the other peers. In Fig. 3, the Vivaldi distances of this

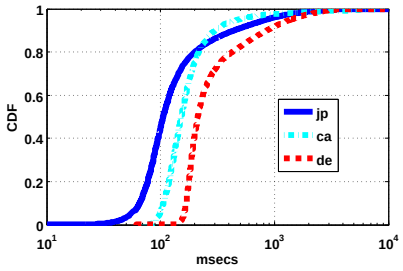


Fig. 3. Vivaldi distances for different countries. Origin peer is in the US.

US peer to peers in Japan, Canada, and Germany are plotted. Surprisingly, in most of the cases, it is not the peers in Canada that appear closer to peers in the US, but the peers in Japan. Thus, based on Vivaldi distance, a peer located in the US would often prefer a Japanese peer over a Canadian peer.

The CDFs of the coordinate distances of a peer based in Germany have a shape similar to the CDFs of the NRTTs shown in Fig. 2.

4.2 Height

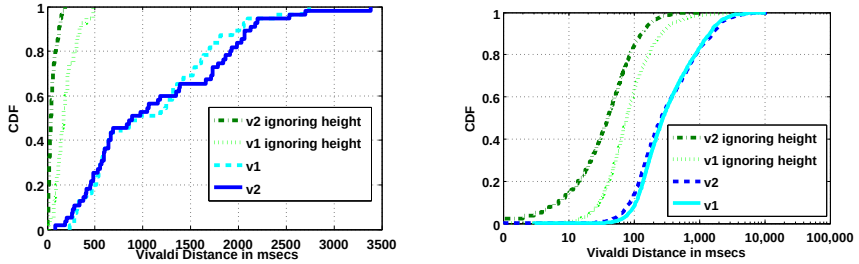
The height value in the network coordinates should reflect the latency introduced by buffers on the last mile toward the end-user. The usage of the height is necessary in order to not distort the coordinate system by large latencies introduced on ADSL lines. We extracted all 112 peers from our dataset that are customers of the provider France Télécom and that are located in Nice / France. The propagation delay between any pair of those clients is in the order of a few milliseconds.

The Vivaldi distance between pairs of those peers is very short if the height value is discarded given the small geographic distances between the hosts. Considering the height, however, they can be far away from one to another. In Fig. 4(a), the CDF of the pairwise distances is plotted. We see that the latency introduced by the ADSL links is completely reflected in the height value and not in the coordinates. For version 2, the results are even better than for version 1, the distance ignoring the height is only of 38 milliseconds in median.

Figure 4(b) shows the pairwise Vivaldi distances of all 12,438 Azureus peers in France. Again, the distances without the height do reflect the geographic distances, whereas the distances including the height are of one order of magnitude larger due to the queuing delay on the last mile.

4.3 Visual Check of the Network Coordinates

Since a network coordinate system assigns coordinates based on the measured latency between the hosts, one should expect to see clusters of peers when plotting the coordinates that correspond to peers on different continents. Theses clusters

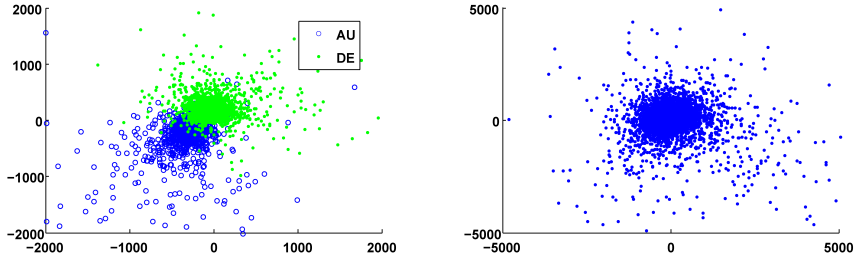


(a) Clients of France Télécom in Nice

(b) All clients in France

Fig. 4. CDF of the pair wise distances of Azureus clients, with and without considering the height value of the coordinates.

should be separated by gaps, the oceans. In Fig. 5(a), we plotted the network coordinates version 1 (without the height) of the German and the Australian peers. There is a strong overlap between them, no clear separation and no gap. This is the same for other pairs of countries plotted together. In Fig. 5(b), the peers' coordinates of all countries are plotted. It is not possible to distinguish between countries or even continents. Geographical distances are not at all represented.



(a) Germany and Australia: overlap, no separation, no gap.

(b) All countries: no distinction possible.

Fig. 5. Azureus network coordinates version 1

These findings are in contradiction to Ledlie et al. [5] who claim that embedding the Internet, which interconnects peers across the globe (the Earth), into an Euclidean space works well due to the fact that traffic between Europe and Asia is routed via the US. The same paper also states that the peers of different continents (Asia, Europe, and North America) cluster together in the network coordinate space, which is in clear disaccord to our findings. In a technical report [16] by the same group, snapshots are presented (Figure 11) of the coordinates of the peers they run on PlanetLab where three clusters representing the continents are distinguishable. We believe that such results can be obtained on PlanetLab but not with peers that are connected via ADSL.

4.4 Which peer to choose?

The classical use of an Internet coordinate system is to choose the peers from which to download from. To check if the coordinate system implemented in Azureus fulfills that request, we set up a very simple experiment. We dumped the Vivaldi coordinates of 2 peers (running on 2 machines in the same LAN) at Eurécom, 2 peers (running on 2 machines in the same LAN) at the University of Mannheim, and one connected via a France Télécom ADSL line every 5 minutes for 9 days, starting on February 25, 2008. Using the coordinates, the peers located in Mannheim, respectively Eurécom, should be able to choose the other peer in the same LAN.

In the following, we computed the Vivaldi distances between those peers (Table 3). When using network coordinates, the distance values for the different pairs of peers make these pairs practically indistinguishable, or, as in the case of version 2, make two peers in Eurécom / Mannheim look closer to each other than peers that are adjacent.

Table 3. Vivaldi distances between two peers. The first column indicates the location of the two peers, the second column indicates the NRTT between them which is stable.

peers	NRTT	version 1			version 2		
	ping	avg.	st.dev.	max	avg.	st.dev.	max
ma 1 / ma 2	0	133	117	1,208	121	56	336
eur 1 / eur 2	0	182	204	2,606	140	72	414
ma 1 / eur 2	39	158	120	786	95	45	302
ma 2 / eur 1	39	144	171	2,563	167	71	390
eur 1 / adsl	70	177	126	1,066	108	37	241
ma 1 / adsl	70	179	121	981	106	37	269

5 Conclusion and Outlook

We have studied the Vivaldi network coordinate system currently implemented in Azureus and evaluated its possibilities and limitations. We saw that the latencies estimated using Vivaldi network coordinates exhibit a high correlation with the round trip times at application layer (ARTT) and to a lesser degree with the round trip times at network layer (NRTT).

In general, the Vivaldi coordinates are not suitable for selecting close-by (geographically or within same ISP) peers: The round trip time is composed of three elements: propagation delay, transmission delay, and queuing delay. As many peers are connected to the Internet via ADSL, the queuing delay of the ADSL access link can dominate the round trip time and hide the contribution of the geographical distance completely, which is reflected in the propagation delay. Extremely long round trip times of several seconds are a strong indication for heavily loaded ADSL links and not for a very large distance between those peers.

Due to this fact, groups of peers in geographical proximity, e.g. same country or ISP, cannot be determined using Vivaldi coordinates.

For an interesting approach to locate "close-by" peers, we refer the reader to recent work of Choffnes and Bustamente [17] who developed an Azureus plug-in called Ono. The plug-in builds a coordinate system based on the measurements performed to several landmarks which are edge servers of the Akamai and Limelight CDN networks.

Given this limitation, Vivaldi coordinates are still very useful for peer selection whenever the round trip time has an impact on the performance, as is the case in query routing and when downloading content via TCP connections.

Some of our traces, the dataset called "Mannheim" (cf. Sect. 4), are available under `http://www.eurecom.fr/~btroup/vivaldiazureus/`.

References

1. Cox, R., Dabek, F., Kaashoek, F., Li, J., Morris, R.: Vivaldi: A Decentralized Network Coordinate System. In: Proceedings of SIGCOMM. (2004)
2. Ng, E., Zhang, H.: Predicting Internet network distance with Coordinates-Based Approaches. In: Proceedings of INFOCOM. (2002)
3. Tang, L., Crovella, M.: Virtual Landmarks for the Internet. In: Proceedings of the Internet Measurement Conference (IMC). (2003)
4. Shavitt, Y., Tankel, T.: Big-Bang Simulation for Embedding Network Distances in Euclidean Space. In: Proceedings of INFOCOM. (2003)
5. Ledlie, J., Gardner, P., Seltzer, M.: Network Coordinates in the Wild. In: 4th USENIX Symposium on Networked Systems Design & Implementation. (2007) 299–311
6. Maymounkov, P., Mazieres, D.: Kademlia: A Peer-to-Peer Information System Based on the XOR Metric. In: Proceedings of the 1^{st} International Workshop on Peer-to-Peer Systems (IPTPS). (2002) 53–65
7. Azureus: `http://azureus.sourceforge.net/`.
8. Overnet: `http://www.overnet.org/`.
9. E-Mule: `http://www.emule-project.net/`.
10. A-Mule: `http://www.amule.org/`.
11. Holz, T., Steiner, M., Dahl, F., Biersack, E.W., Freiling, F.: Measurements and Mitigation of Peer-to-Peer-based Botnets: A Case Study on Storm Worm. In: First Usenix Workshop on Large-scale Exploits and Emergent Threats (LEET). (2008)
12. Steiner, M., En-Najjary, T., Biersack, E.W.: A Global View of KAD. In: Proceedings of the Internet Measurement Conference (IMC). (2007)
13. Postel, J.: Transmission Control Protocol – Protocol Specification. Request for Comments (Standard) RFC 793, Information Sciences Institute, USC (1981)
14. Maxmind: `http://www.maxmind.com/`.
15. Dischinger, M., Haeberlen, A., Gummadi, K.P., Saroiu, S.: Characterizing Residential Broadband Networks. In: Proceedings of the Internet Measurement Conference (IMC). (2007)
16. Ledlie, J., Gardner, P., Seltzer, M.: Network Coordinates in the Wild. Technical report, University of Harvard, Cambridge, MA, US (2007)
17. Choffnes, D.R., Bustamante, F.E.: Taming the Torrent: A practical approach to reducing cross-ISP traffic in P2P systems. In: Proceedings of ACM SIGCOMM. (2008)