Sebastian Kaune, Matthias Wählisch, Konstantin Pussep:
Modeling and Tools for Network Simulation: Modeling the Internet Delay Space and its Application in Large Scale P2P Simulations. In: Klaus Wehrle, Mesut Günes, James Groß, p. 427--446, Springer, July 2010. ISBN 978-3-642-12330-6.

19. Modeling the Internet Delay Space and its Application in Large Scale P2P Simulations

Sebastian Kaune (Technische Universität Darmstadt)
Matthias Wählisch (Freie Universität Berlin & HAW Hamburg)
Konstantin Pussep (Technische Universität Darmstadt)

19.1 Introduction

The peer-to-peer (P2P) paradigm has greatly influenced the design of Internet applications nowadays. It gained both user popularity and significant attention from the research community, aiming to address various issues arising from the decentralized, autonomous, and the self-organizing nature of P2P systems [379]. In this regard, quantitative and qualitative analysis at large scale is a crucial part of that research. When evaluating widely deployed peer-to-peer systems an analytical approach becomes, however, ineffective due to the large number of simplifications required. Therefore, conclusions about the real-world performance of P2P systems can only be drawn by either launching an Internet-based prototype or by creating a simulation environment that accurately captures the major characteristics of the heterogeneous Internet, e.g. round-trip times, packet loss, and jitter. Running large scale experiments with prototypes is a very challenging task due to the lack of sufficiently sized testbeds. While PlanetLab [36] consists of about 800 nodes, it is still too small and not diverse enough [434] to provide a precise snapshot for a qualitative and quantitative analysis of a P2P system. For that reason, simulation is often the most appropriate evaluation method.

Internet properties, and especially their *delay* characteristics, often directly influence the performance of protocols and systems. In delay-optimized overlays, for instance, *proximity neighbor selection* (PNS) algorithms select the closest node in the underlying network from among those that are considered equivalent by the routing table. The definition of closeness is typically based on round-trip time (RTT). In addition, many real time streaming systems (audio and video) have inherent delay constraints. Consequently, the Internet *end-to-end delay* is a significant parameter affecting the user's satisfaction with the service. Therefore, in order to obtain accurate results, simulations must include an adequate model of the Internet delay space.

We begin by discussing the factors that may affect the Internet end-to-end delay in Section 19.2. Section 19.3 gives an overview on state-of-the art Internet delay models. In Section 19.4 and 19.5, we present background

information and details on a novel delay model, which we evaluate in Section 19.6. Concluding remarks are given in Section 19.7.

19.2 End-to-end Delay and Its Phenomena

In order to accurately model the Internet delay characteristics, the influencing entities and their inherent phenomena must be identified. We define the term *Internet end-to-end delay* as the length of time it takes for a packet to travel from the source host to its destination host. In more detail, this packet is routed to the destination host via a sequence of intermediate nodes. The Internet end-to-end delay is therefore the sum of the delays experienced at each hop on the way to the destination. Each such delay in turn consists of two components, a fixed and a variable component [68]. The *fixed* component includes the transmission delay at a node and the propagation delay on the link to the next node. The *variable* component, on the other side, includes the processing and queuing delays at the node.

Normally, end-to-end delays vary over time[410]. We denote this *delay variation* as *end-to-end delay jitter*. According to [126], there are three major factors that may affect the end-to-end delay variation: queueing delay variations at each hop along the Internet path; intra-domain multi-path routing, and inter-domain route alterations.

Thus, the main challenges in creating a Internet delay space model can be summarized as follows:

- The model must be able to predict lifelike delays and jitter between a given pair of end-hosts.
- The computation of delays must scale with respect to time.
- The model must have a compact representation.

We argue that the first requirement is subject to the geographical position of the sender and the receiver. First, the minimal end-to-end delay between two hosts is limited by the propagation speed of signals in the involved links which increases proportionally with the link length. Second, the state of the Internet infrastructure varies significantly in different countries. As long-term measurement studies reveal (cf. Sec. 19.4), jitter and packet loss rates are heavily influenced by the location of participating nodes. For example, the routers in a developing country are more likely to suffer from overload than those in a more economically advanced country.

Asymmetric Delays

The Internet end-to-end delay refers to the packet travel time from a source to its receiver. This one-way delay (OWD) will typically be calculated

by halving the measured RTT between two hosts, which consists of the forward and reverse portion. Such an estimation most likely holds true, if the path is symmetric. Symmetric paths, however, are not an obvious case. Radio devices, for instance, may experience inhomogeneous connectivity depending on coverage and interferences. Home users attached via ADSL possess inherently different up- and downstream rates. Independent of the access technology in use, Internet routing is generally *not symmetric*, i.e., intermediate nodes traversed from the source to the receiver may differ from the reverse direction. In the mid of 1996, Paxson revealed that 50 % of the virtual Internet paths are asymmetric [357]. Nevertheless, implications for the corresponding delays are not evident. Although router-level paths may vary, the forward and reverse OWD can be almost equal due to similar path lengths, router load etc.

Internet delay asymmetry has been studied in [354]. The authors show that an asymmetric OWD implies different forward and reverse paths. However, unequal router-level paths do not necessarily imply asymmetric delays [354]. An asymmetric OWD could be mainly identified for commercial networks compared to research and education backbones. It is worth noting that the end-to-end delay between two hosts within different autonomous systems (ASes) is significantly determined by the intra-AS packet travel time [512]. Combining the observations in [354] and [512] thus suggest that in particular delays between hosts located in different provider domains are poorly estimated by the half of RTT.

The approximation of the OWD by RTT/2 may over- or underestimate the delay between two hosts. In contrast to the RTT, measuring the OWD is a more complex and intrinsic task as it requires the dedicated cooperation of the source as well as its receiver [416], [480]. Consequently, hosts cannot instantaneously discover the OWD. Protocols and applications therefore use the RTT, e.g., P2P applications while applying this metric for proximity neighbor selection. The modeling process of network structures which include end-to-end delays should be aware of the asymmetric delay phenomena. Neglecting this Internet property seems reasonable when deployment issues allow for the simplification, or it is common practice in the specific context. Otherwise, the approximation is unreasonable.

In the following sections of this chapter, we will focus on geometric schemes to model the delay space. These approaches calculate the packet travel time based on the Euclidean distance of artificial network coordinates. Obviously, such models cannot account for delay asymmetry as the Euclidean distance between two points is symmetric per definition. Further, we often use the term delay as synonym for end-to-end or *one-way delay*.

19.3 Existing Models in Literature

Currently, there are four different approaches to obtaining an Internet delay model: analytical functions, the king method, topology generators, and Euclidean embedding. In this section, we will briefly discuss each of those approaches.

Analytical *function.* The simplest approach to predict delay is to randomly place hosts into an two-dimensional Euclidean space. The delay is then computed by an analytical function that uses as an input the distance between any two hosts, for example, the Euclidean distance. While this approach requires only simple run-time computations and does not introduce any memory overhead, it has one major drawback: it neglects the geographical distribution and locations of hosts on earth, which are needed for both the realistic modeling of lifelike delays (i) and jitter (ii).

King method. The second approach uses the King tool [247] to compute the all-pair end-to-end delays among a large number (typically dozens of thousands) of globally distributed DNS servers. In more detail, each server is located in a distinct domain, and the measured delays therefore represent the Internet delay space among the edge networks [513]. Due to the quadratic time requirement for collecting this data, the amount of measured data is often limited. For example, [247] provides a delay matrix with 1740 rows/columns. This is a non-trivial amount of measurement data to obtain, but might be too less for huge P2P systems consisting over several thousands of nodes. To tackle this issue, a delay synthesizer may be used that uses the measured statistical data as an input in order to produce Internet delay spaces at a large scale [513]. Nevertheless, this synthesizer only produces static delays and neglect the delay variation.

Topology generators. The third approach is based on using artificial link delays assigned by topology generators such as Inet [232] or GT-ITM [511]. This scheme initially generates a topology file for a predefined number of nodes n. A strategy for the final computation of the end-to-end delay depends on the specific scenario and should consider two issues: (a) on-demand vs. pre-computation and (b) the single-source path (SSP) vs. all-pair shortest path (ASP) problem[1]. In contrast to an on-demand calculation, a pre-calculation may reduce the overall computational costs if delays are required several times, but increases the memory overhead. The ASP problem, which causes high computational power and squares the memory overhead to $O(n^2)$, should be solved in the case that delays between almost all nodes are needed. It is sufficient to separately calculate the SSP, if only a small subset of nodes will be analyzed.

[1] We refer to the SSP and ASP problem as example for solving a routing decision for some or all nodes.

Model	Computation cost	Memory overhead	Comment
Analytical function	low	$O(1)$	static delays neglects geographical pos.
King method	low	$O(n^2)$	static delays very high precision complicated data acquisition
Topology generators (pre-computation)	low	$O(n^2)$	static delays neglects geographical pos.
Topology generators (on-demand)	very high (Dijkstra's SSP)	low	static delays neglects geographical pos.
Euclidean embedding	low	$O(n)$	data freely available

Table 19.1: Different approaches for modeling the Internet delay space. The number of end-hosts is denoted by n.

Euclidean embedding. The fourth approach is based on the data of Internet measurement projects, e.g. Surveyor [450], CAIDA [85], and AMP [25], which are freely available. These projects typically perform active probing up to a million destination hosts, derived from a small number of globally distributed monitor hosts. This data is used as an input to generate realistic delay by embedding hosts into a multi-dimensional Euclidean space [168].

Table 19.1 gives an overview about the properties of the aforementioned approaches. Unfortunately, none of them considers realistic delay and jitter based on the geographical position of hosts. That is, these approaches aim to predict static delays, either the average or minimum delay between two hosts. Furthermore, most of them do not accurately reflect delay characteristics caused by different geographical regions of the world. This issue can, however, highly influence the performance of P2P systems, as we will see in Section 19.5.3. Only the Euclidean embedding seems to be an optimal tradeoff between computational costs and memory overhead.

In the remainder of this chapter, we therefore present an alternative approach of obtaining end-to-end delays that fulfills the requirements stated in the previous section. It exploits the compact and scalable representation of hosts in an Euclidean embedding, whilst considering the geographical position of hosts to calculate delays and lifelike jitter. This approach is based on rich data from two measurement projects as input.

19.4 Data from two Internet Measurement Projects

This section provides background information on the measured Internet delay data we use in our model. Firstly, we use the measurement data of the CAIDA's macroscopic topology probing project [85]. This data contains a large volume of RTT measurements taken between 20 globally distributed

monitor hosts[2] and nearly 400,000 destination hosts. Within this project, each monitor actively probes every host stored in the so-called destination list by sending ICMP [371] echo-requests. This lists account for 313,471 hosts covering the routable IPv4 space, alongside 58,312 DNS clients. Each monitor-to-destination link is measured 5-10 times a month, resulting in an overall amount of 40 GB of measurement data. As an example, Fig. 19.1 plots the data of August 2007 in relation to the geographical distance between each monitor host and its destinations. Both, the *geographical locations* of the monitors and the destination hosts are determined by *MaxMind* GeoIP service[3] [309]. It can be observed that there is a proportionality of the RTT to the length of the transmission medium. The 'islands' at 8000 - 12000 km and 300 - 400 ms RTT arises from countries in Africa and South Asia.

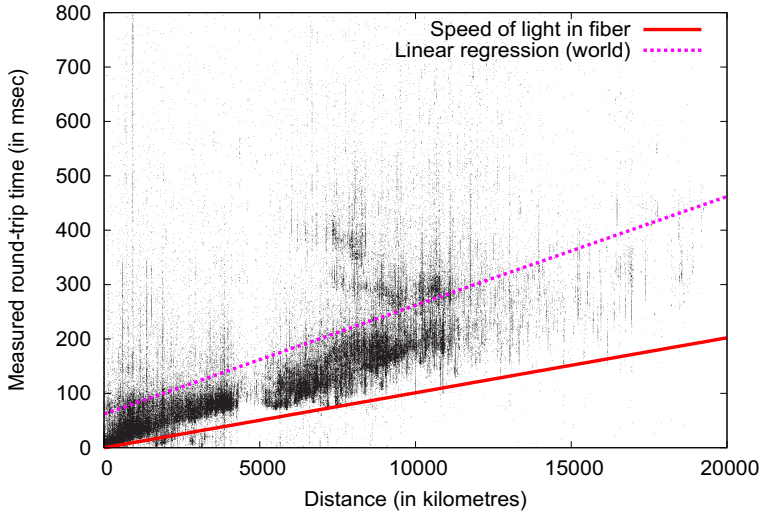

Fig. 19.1: The measured *round-trip* times in relation to the geographical distance in August 2007

To study the changes of delay over time, we additionally incorporate the data of the PingER project [463]. This project currently has more than 40 monitoring sites in 20 countries and about 670 destination sites in 150 countries. This number of monitor hosts is double than that of the CAIDA project, whereas the amount of remote sites is by order of magnitudes smaller. Nevertheless, the RTT for one monitor-to-destination link is measured up to 960 times a day, in contrast to 5-10 times per month by the CAIDA project.

[2] For more information about the monitor hosts, see http://www.caida.org/projects/ark/statistics/index.xml

[3] The obviously impossible RTT values below the propagation time of the speed of light in fiber can be explained by a false positioning through MaxMind.

As seen later on, this allows us to accurately predict the inter-packet delay variation between any two hosts located in different countries or continents.

19.5 Model

This section details our model that aims to realistically predict end-to-end delays between two arbitrary hosts chosen from a predefined host set. This model approximates the OWD between two hosts by halving the measured RTTs as obtained from the above mentioned measurement projects. However, we are aware that this approach may over- or underestimate the actual OWD in reality (cf. Sec 19.2). Nevertheless, the obtained delays are non-static, and consider the geographical location of both the source and destination host. Further, the model properties in terms of computation and memory overhead are given.

19.5.1 Overview

We split up the modelling of delay into a two-part architecture. The first part computes the *minimum* one-way delay between two distinct hosts based on the measured round-trip time samples of CAIDA, and is therefore static. The second part, on the other hand, is variable and determines the jitter.

Thus, the OWD between two hosts $\mathcal{H}_1$ and $\mathcal{H}_2$ is given by

$$delay(\mathcal{H}_1, \mathcal{H}_2) = \frac{RTT_{min}}{2} + jitter. \tag{19.1}$$

Fig. 19.2 gives an overview of our model. The *static part* (top left) generates a set of hosts from which the simulation framework can choose a subset from. More precisely, this set is composed of the destination list of the CAIDA measurement project. Using the MaxMind GeoIP database, we are able to look up the IP addresses of these hosts and find out their geographic position, i.e., continent, country, region, and ISP. In order to calculate the minimum delay between any two hosts, the Internet is modelled as a multidimensional Euclidean space $\mathcal{S}$. Each host is then mapped to a point in this space so that the minimum round-trip time between any two nodes can be predicted by their Euclidean distance.

The *random part* (top right), on the other hand, determines the inter-packet delay variation of this minimum delay; it uses the rich data of the PingER project to reproduce end-to-end link jitter distributions. These distributions can then be used to calculate random jitter values at simulation runtime.

Basically, both parts of our architecture require an offline computation phase to prepare the data needed for the simulation framework. Our overall

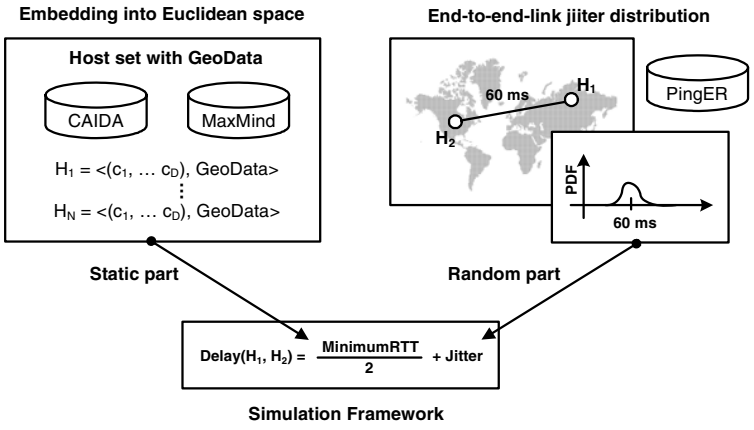

Fig. 19.2: Overview of our delay space modeling techniques

goal is then to have a very compact and scalable presentation of the underlay at simulation runtime without introducing a significant computational overhead. In the following, we describe each part of the architecture in detail.

19.5.2 Part I: Embedding CAIDA hosts into the Euclidean Space

The main challenge of the first part is to position the set of destination hosts into a multidimensional Euclidean space, so that the computed minimum round-trip times approximate the measured distance as accurately as possible. To do so, we follow the approach of [335] and apply the technique of global network positioning. This results in an optimization problem of minimizing the sum of the error between the measured RTT and the calculated distances.

In the following, we denote the coordinate of a host $\mathcal{H}$ in a D-dimensional coordinate space $\mathcal{S}$ as $c_{\mathcal{H}} = (c_{\mathcal{H},1}, ..., c_{\mathcal{H},D})$. The measured round-trip time between the hosts $\mathcal{H}_1$ and $\mathcal{H}_2$ is given by $d_{\mathcal{H}_1\mathcal{H}_2}$ whilst the computed distance $\hat{d}_{\mathcal{H}_1\mathcal{H}_2}$ is defined by a distance function that operates on those coordinates:

$$\hat{d}_{\mathcal{H}_1\mathcal{H}_2} = \sqrt{(c_{\mathcal{H}_1,1} - c_{\mathcal{H}_2,1})^2 + ... + (c_{\mathcal{H}_1,D} - c_{\mathcal{H}_2,D})^2}. \tag{19.2}$$

As needed for the minimization problems described below, we introduce a weighted error function $\varepsilon(\cdot)$ to measure the quality of each performed embedding:

$$\varepsilon(d_{\mathcal{H}_1\mathcal{H}_2}, \hat{d}_{\mathcal{H}_1\mathcal{H}_2}) = \left(\frac{d_{\mathcal{H}_1\mathcal{H}_2} - \hat{d}_{\mathcal{H}_1\mathcal{H}_2}}{d_{\mathcal{H}_1\mathcal{H}_2}} \right)^2. \tag{19.3}$$

Basically, this function calculates the squared error between the predicted and measured RTT in a weighted fashion and has been shown to produce accurate coordinates, compared to other error measures [335].

At first, we calculate the coordinates of a small sample of N hosts, also known as *landmarks* $\mathcal{L}_1$ to $\mathcal{L}_N$. A precondition for the selected landmarks is the existence of measured round-trip times to each other. In our approach, these landmarks are chosen from the set of measurement monitors from the CAIDA project, since these monitors fulfill this precondition. In order to achieve a good quality of embedding, the subset of N monitors must, however, be selected with care.

Formally, the goal is to obtain a set of coordinates $c_{\mathcal{L}_1}, ..., c_{\mathcal{L}_N}$ for the selected N monitors. These coordinates then serve as reference points with which the position of any destination host can be oriented in $\mathcal{S}$. To do so, we seek to minimize the following objective function f_{obj1}:

$$f_{obj1}(c_{\mathcal{L}_1}, ..., c_{\mathcal{L}_N}) = \sum_{i=1|i>j}^{N} \varepsilon(d_{\mathcal{L}_i\mathcal{L}_j}, \hat{d}_{\mathcal{L}_i\mathcal{L}_j}). \tag{19.4}$$

There are many approaches with different computational costs that can be applied [295], [335]. Recent studies have shown that a five dimensional Euclidean embedding approximates the Internet delay space very well [397]. Therefore, we select $N(=6)$ nodes out of all available monitors using the maximum separation method[4] [168]. For this method, we consider, however, only the minimum value across the samples of inter-monitor RTT measurements.

In the second step, each destination host is iteratively embedded into the Euclidean space. To do this, round-trip time measurements to all N monitor hosts must be available. Similarly to the previous step, we take the minimum value across the monitor-to-host RTT samples. While positioning the destination hosts coordinate into $\mathcal{S}$, we aim to minimize the overall error between the predicted and measured monitor-to-host RTT by solving the following minimization problem f_{obj2}:

$$f_{obj2}(c_{\mathcal{H}}) = \sum_{i=1}^{N} \varepsilon(d_{\mathcal{H}\mathcal{L}_i}, \hat{d}_{\mathcal{H}\mathcal{L}_i}). \tag{19.5}$$

Because an exact solution of this non-linear optimization problem is very complex and computationally intensive, an approximative solution can be found by applying the generic *downhill simplex algorithm* of Nelder and Mead [230].

[4] This method determines the subset of N monitors out of all available monitors which produces the maximum sum for all inter-monitor round-trip times.

19.5.3 Part II: Calculation of Jitter

Since the jitter constitutes the variable part of the delay, a distribution function is needed that covers its lifelike characteristics. Inspection of the measurement data from the PingER project shows that this deviation clearly depends on the geographical region of both end-hosts. Table 19.2 depicts an excerpt of the two way-jitter variations of end-to-end links between hosts located in different places in the world. These variations can be monthly accessed on a regional-, country-, and continental level [463]. We note that these values specify the *interquartile range* (iqr) of the jitter for each end-to-end link constellation. This range is defined by the difference between the upper (or third) quartile Q_3 and the lower (or first) quartile Q_1 of all measured samples within one month. The remarkably high iqr-values between Africa and the rest of the world are explained by the insufficient stage of development of the public infrastructure.

To obtain random jitter values based on the geographical position of hosts, for each end-to-end link constellation we generate a log-normal distribution[5] with the following probability distribution function:

$$\mathrm{f}(x;\mu,\sigma) = \begin{cases} \frac{1}{\sqrt{2\pi}\sigma x} \exp\left(-\frac{1}{2}\left(\frac{\ln x-\mu}{\sigma}\right)^2\right) & \text{if } x > 0 \\ 0 & \text{otherwise.} \end{cases} \tag{19.6}$$

The main challenge is then to identify the parameters μ (mean) and σ (standard deviation) by incorporating the measurement data mentioned above. Unfortunately, both values cannot be obtained directly from PingER. That is, we are in fact able to determine the expectation value of each constellation, which is given by the difference between the average RTT and the minimum RTT. Both values are also measured by the PingER project, and are available in the monthly summary reports, too. The variance or standard deviation is, however, missing.

For this reason, we formulate an optimization problem that seeks to find a parameter configuration for μ and σ having two different goals in mind. First, the chosen configuration should minimize the error between the measured inter quartile range $\mathrm{iqr_m}$ and $\mathrm{iqr}(X)$ which is generated by the log-normal distribution. Second, it should also minimize the measured and generated expectation, $\mathrm{E_m}$ and $\mathrm{E}(X)$ respectively. Formally, this optimization problem is given by

$$f_{error} = \left(\frac{\mathrm{E}(X)-\mathrm{E_m}}{\mathrm{E_m}}\right)^2 + \left(\frac{\mathrm{iqr}(X)-\mathrm{iqr_m}}{\mathrm{iqr_m}}\right)^2. \tag{19.7}$$

[5] In [168], it is shown based on real measurements that jitter values can be approximated by a log-normal distribution.

	Europe	Africa	S. America	N. America	Asia
Europe	1.53	137.14	3.07	1.29	1.19
Africa	26.91	78.17	3.69	31.79	1.11
S. America	14.17	69.66	13.14	10.78	14.16
N. America	2.02	73.95	3.63	0.96	1.33
Oceania	4.91	86.28	4.19	1.31	2.03
Balkans	1.83	158.89	3.89	1.43	1.25
E. Asia	1.84	114.55	3.02	1.38	0.87
Russia	2.29	161.34	4.79	2.53	1.59
S. Asia	7.96	99.36	8.99	16.48	7.46
S.E. Asia	0.86	83.34	4.43	13.36	1.27
Middle East	9.04	120.23	11.39	10.87	10.20

Table 19.2: End-to-end link inter-packet delay variation in msec (January 2008).

where $\mathrm{E}(X) = e^{\mu+\sigma^2/2}$ and $\mathrm{iqr}(X) = Q_3 - Q_1$ as described above. To solve this, we apply the downhill simplex algorithm [230]. Observation of measurement data shows that the iqr-values are usually in the range of 0 to 20 milliseconds[6]. With respect to this, the three initial solutions are set to $(\mu = 0.1, \sigma = 0.1)$, $(\mu = 0.1, \sigma = 5)$, and $(\mu = 5, \sigma = 0.1)$, because these parameters generate random jitter values fitting this range exactly. The minimization procedure iterates then only 100 times to obtain accurate results.

We note that the obtained values for μ and σ describe the distribution of the two-way jitter for a specific end-to-end link constellation. The one-way jitter is then obtained by dividing the randomly generated values by two. Further, each end-to-end link constellation is *directed* from a geographical region. For example, the delay variation of a packet that travels from Europe to Africa is significantly higher than the one from Africa to Europe (cf. Tab. 19.2). By using two directed end-to-end link constellations, one starting from Europe and the other one starting from Africa, we are able to reflect this asymmetry.

19.5.4 Algorithm and Memory Overhead

In this section, we briefly describe the properties of our model in terms of computational costs and storage overhead. These properties are of major importance since they significantly influence the applicability of the model in large scale simulations.

First of all, the embedding of all hosts n into a D-dimensional Euclidean space has a scalable representation of $O(n)$ while it adequately preserves the properties of the data measured by the CAIDA project. Since the process

[6] Africa constitutes a special case. For this, we use another initial configuration as input for the downhill simplex algorithm.

involved in obtaining this representation is complex and computationally expensive, it is typically done once. The resulting data can be reused for each simulation run, e.g., in terms of an XML file. In order to obtain the minimum delay between any two hosts in this embedding, the evaluation of the distance function takes then $O(D)$ time which is negligible.

The calculation of the jitter parameters of μ and σ for each possible end-to-end link constellation is also done once, either before the simulation starts or offline. Thus, similar to the pre-computation of the host coordinates, this process does not introduce any computational overhead into the actual simulation process. Nevertheless, the storage of the both parameters μ and σ takes at first sight a quadratic overhead of $O(n^2)$. Due to the fact that the amount of regions, countries and continents is limited, the required amount of memory is, however, negligible. For example, the processing of the data provided in the PingER summary report of January 2008 result in 1525 distinct link constellations. For each of them, the two parameters μ and σ must be precomputed and stored resulting in a overall storage overhead of $(1525 \times 2) \times 4\,\text{bytes} \approx 12\text{kB}$.

19.6 Evaluation

This section describes the setup of our experiments, and any metrics we think significantly influence the performance of P2P systems. We perform a comparative study against three existing approaches for obtaining end-to-end delays: (*i*) the King method, (*ii*) topology generators and (*iii*) analytical function. Our aim is to show that our model realistically reflects the properties of the Internet delay space. To this end, we show that the calculated delay between non-measured end-to-end links is also a suitable presumption compared to the delays that occur in the Internet.

19.6.1 Experimental Setup

The King method serves as a reference point in our analysis because it provides measured Internet delay data among a large number of globally distributed DNS servers. We use the measurement data of [513] collected in October 2005. This matrix contains 3997 rows/columns representing the all-pair delays between IP hosts located in North America, Europe and Asia.

With regard to the topology generators, we are especially interested in the GT-ITM and Inet generators because they are often used in P2P simulations. For GT-ITM, we create a 9090 node transit-stub topology. For Inet, we create a topology for a network size of 10000 nodes. We use the default settings of placing nodes on a 10000 by 10000 plane with 30% of total nodes as degree-one nodes.

As seen in Section 19.4, there is a correlation between the measured RTTs and the geographical distance of peers. In order to obtain an analytical function that reflects this correlation, we perform a least squares analysis so that the sum of the squared differences between the calculated and the measured RTT is minimized. Applying linear regression with this least squares method on the measurement data of 40 GB is, however, hardly possible. Therefore, we classify this data into equidistant intervals of 200 km (e.g. $(0\text{km}, 200\text{km}], (200\text{km}, 400\text{km}] \ldots$), and calculate the median round-trip time of each interval. Finally, linear regression gives us the following estimation for the RTT in milliseconds:

$$f_{world}(d_{a,b}) = 62 + 0.02 * d_{a,b} \tag{19.8}$$

whereas $d_{a,b}$ is the distance between two hosts in kilometers. The delay is then given by $f(d_{a,b})$ divided by two. Fig. 19.3 illustrates this function and the calculated median RTT times of each interval.

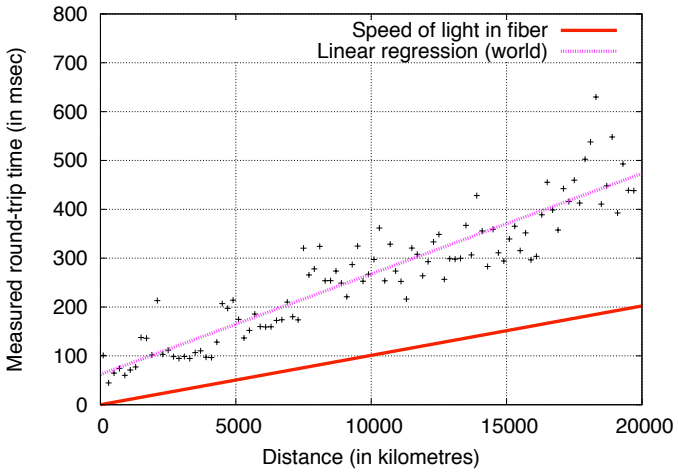

Fig. 19.3: Results of linear regression with least square analysis on CAIDA measurement data.

19.6.2 Metrics

To benchmark the different approaches on their ability to realistically reflect Internet delay characteristics, we apply a set of metrics that are known to significantly influence the performance of P2P systems [513]:

- *Cutoff delay clustering* – In the area of P2P content distribution networks, topologically aware clustering is a very important issue. Nodes are often grouped into clusters based on their delay characteristics, in order to provide higher bandwidth and to speed up access [169]. The underlying delay model must therefore accurately reflect the Internet's clustering properties. Otherwise, analysis of system performance might lead to wrong conclusions.

  To quantify this, we use a clustering algorithm which iteratively merges two distinct clusters into a larger one until a cutoff delay value is reached. In more detail, at first each host is treated as a singleton cluster. The algorithm then determines the two closest clusters to merge. The notion of closeness between two clusters is defined as the average delay between all nodes contained in both cluster. The merging process stops if the delay of the two closest clusters exceeds the predefined cutoff value. Afterwards, we calculate the fraction of hosts contained in the largest cluster compared to the entire host set under study.

- *Spatial growth metric* – In many application areas of P2P systems, such as in mobile P2P overlays, the cost of accessing a data object grows as the number of hops to the object increases. Therefore, it is often advantageous to locate the 'closest' copy of a data object to lower operating costs and reduce response times. Efficient distributed nearest neighbor selection algorithms have been proposed to tackle this issue for growth-restricted metric spaces [22]. In this metric space, the number of nodes contained in the radius of delay r around node p, increases at most by a constant factor c when doubling this delay radius. Formally, let $B_p(r)$ denote the number of nodes contained in a delay radius r, then $B_p(r) \leq c \cdot B_p(2r)$. The function $B_p(r)/B_p(2r)$ can therefore be used to determine the spatial growth c of a delay space.

- *Proximity metric* – In structured P2P overlays which apply proximity neighbor selection (PNS), overlay neighbors are selected by locating nearby underlay nodes [185]. Thus, these systems are very sensitive to the underlying network topology, and especially to its delay characteristics. An insufficient model of the Internet delay space would result in routing table entries that do not occur in reality. This would in turn directly influence the routing performance and conclusions might then be misleading. To reflect the neighborhood from the point of view of each host, we use the $\mathcal{D}(k)$-metric. This metric is defined by $\mathcal{D}(k) = \frac{1}{|N|} \sum_{p \in N} d(p,k)$, whereas $d(p,k)$ is the average delay from node p to its k-closest neighbors in the underlying network [297].

19.6.3 Analysis with Measured CAIDA data

Before we compare our system against existing approaches, we briefly show that our delay model produces lifelike delays even though their calculation is divided into two distinct parts.

As an illustration of our results, Fig. 19.4 depicts the measured RTT distribution for the Internet as seen from CAIDA monitors in three different geographical locations, as well as the RTTs predicted by our model. We note that these distributions now contain all available samples to each distinct host, as opposed to the previous section where we only considered the minimum RTT.

First, we observe that our predicted RTT distribution accurately matches the measured distribution of each monitor host. Second, the RTT distribution varies substantially in different locations of the world. For example, the measured path latencies from China to end-hosts spread across the world have a median RTT more than double that of the median RTT measured in Europe, and even triple that of the median RTT measured in the US. Additionally, there is a noticeable commonality between all these monitors regarding to the fact that the curves rise sharply in a certain RTT interval, before they abruptly flatten out. The former fact indicates a very high latency distribution within these intervals, whereas the latter shows that a significant fraction of the real-world RTTs are in the order of 200 ms and above.

In contrast to this, Fig. 19.5 shows the RTT distribution as seen from a typical node of the network when using the topologies generated by Inet and GT-ITM as stated before. When comparing Fig. 19.4 and Fig. 19.5, it can be observed that the real-world RTT distributions significantly differ from the RTT distributions created by the topology generators. In particular, around 10-20% of the real-world latencies are more than double than their median RTT. This holds especially true for the monitor hosts located in Europe and in the US (see Fig. 19.4). Topology generators do not reflect this characteristic. Additionally, our experiments showed that in the generated topologies, the RTT distribution seen by different nodes does not significantly vary, even though they are placed in different autonomous subsystems and/or router levels. Thus, current topology generators do not accurately reflect the geographical position of peers, something which heavily influences the node's latency distribution for the Internet.

19.6.4 Comparison to Existing Models

We compare our model (coordinate-based) against existing approaches for obtaining end-to-end delays using the metrics presented before. The reference point for each metric is the all-pair delay matrix received by the King method. We use this because the data is directly derived from the Internet. However, we are aware that this data only represents the delay space among

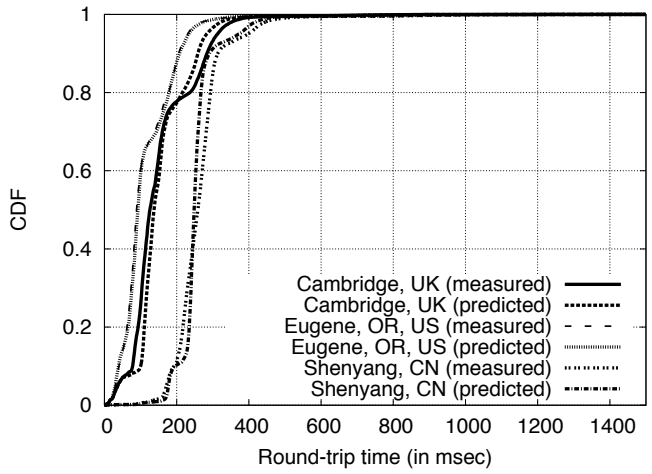

Fig. 19.4: The measured and predicted round-trip time distribution as seen from different locations in the world.

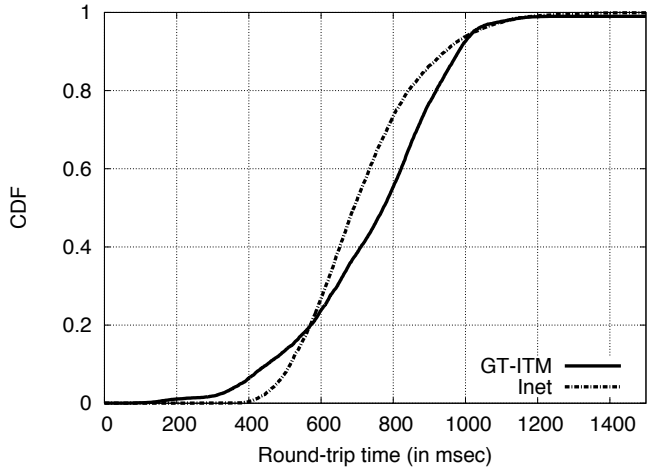

Fig. 19.5: The round-trip time distribution as seen from a typical node generated by topology generators.

the edge networks. To enable a fair comparison, we select, from our final host set, all hosts that are marked as DNS servers in CAIDA's destination list. We only utilize those that are located in Europe, Northern America or Asia. These nodes form the host pool for our coordinate-based model, and the analytical function, from which we chose random sub-samples later on. For the generated GT-ITM topology, we select only stub routers for our experiments to obtain the delays among the edge networks. For the Inet topology, we repeat this procedure for all degree-1 nodes. To this end, we scale the delays derived from both topologies such that their average delays matches the average delay of our reference model. While this process does not affect delay distribution's properties, it alleviates the direct comparison of results.

The results presented in the following are the averages over 10 random sub-samples of each host pool whereas the sample size for each run amounts to 3000 nodes[7].

We begin to analyse the cluster properties of the delay spaces produced by each individual approach. Fig. 19.6 illustrates our results after applying the clustering algorithm with varying cutoff values. It can be observed that for the reference model, our approach , and the distance function, the curves rise sharply at three different cutoff values. This indicates the existence of three major clusters. By inspecting the geographical origin of the cluster members of the latter two models, we find that these clusters exactly constitute the following three regions: Europe, Asia and North America. Further, the three cutoff values of the analytical function are highly shifted to the left, compared to the values of the reference model. Nevertheless, the basic cluster properties are preserved. The curve of our delay model most accurately follows the one of the reference model, but it is still shifted by 10-20 ms to the left. Finally, both topology generated delays do not feature any clear clustering property. This confirms the findings that have already been observed in [513].

To analyse the growth properties of each delay space, we performed several experiments each time incrementing the radius r by one millisecond. Fig. 19.7 depicts our results. The x-axis illustrates the variation of the delay radius r whereas the y-axis shows the median of all obtained $B_p(2r) \,/\, B_p(r)$ samples for each specific value of r. Regarding the reference model, it can be seen that the curves oscillates two times having a peak at delay radius values 20 ms and 102 ms. Also, our coordinate-based approach and the analytical function produces these two characteristic peaks at 26 ms and 80 ms, and 31 ms and 76 ms respectively[8].

In all of the three mentioned delay spaces, the increase of the delay radius firstly covers most of the nodes located in each of the three major clusters. Afterwards, the spatial growth decreases as long as r is high enough to cover

[7] It is shown in [513] that the properties we are going to ascertain by our metrics are independent of the sample size. Thus, it does not matter if we set it to 500 or 3000 nodes.

[8] The minimum delay produced by the analytical function is 31 ms, no matter the distance. This is why there are no values for the first 30 ms of r.

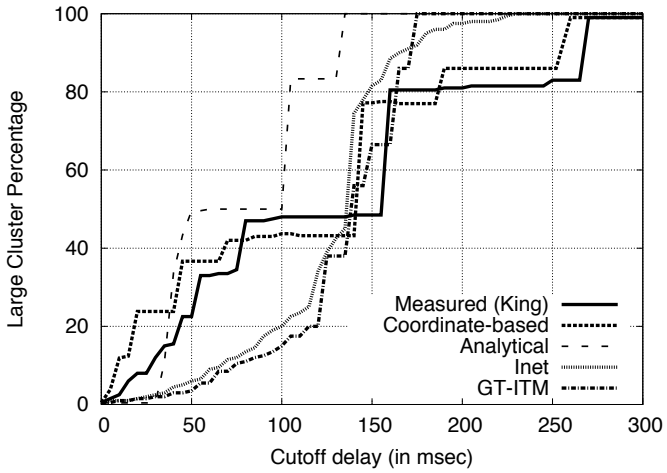

Fig. 19.6: Simulation results for cutoff delay clustering.

nodes located in another major cluster. Lastly, it increases again until all nodes are covered, and the curves flatten out. The derived growth constant for this first peak of the analytical function is, however, an order of magnitude higher than the constants of the others. This is clearly a consequence of our approximation through linear regression. Since this function only represents an average view on the global RTTs, it cannot predict lifelike delays with regard to the geographical location of peers. Nevertheless, this function performs better than both topology generated delay spaces. More precisely, none of both reflect the growth properties observed by our reference delay space.

The experiments with the $D(k)$-metric confirm the trend of our previous findings. The predicted delays of our coordinate-based model accurately matches the measured delays of the reference model. Fig. 19.8 illustrates the simulation results. While varying the number of k (x-axis), we plot the delay derived by the $D(k)$-function over the average to all-node delay. Whilst especially the measured delays and the one predicted by our model show the noticeable characteristic that there are a few nodes whose delay are significantly smaller than the overall average, the topology generated delays do not resemble this. As a consequence, it is likely that the application of PNS mechanisms in reality will lead to highly different results when compared to the ones forecasted with GT-ITM or Inet topologies. The analytical function, on the other hand, performs significantly better than the topology generators,

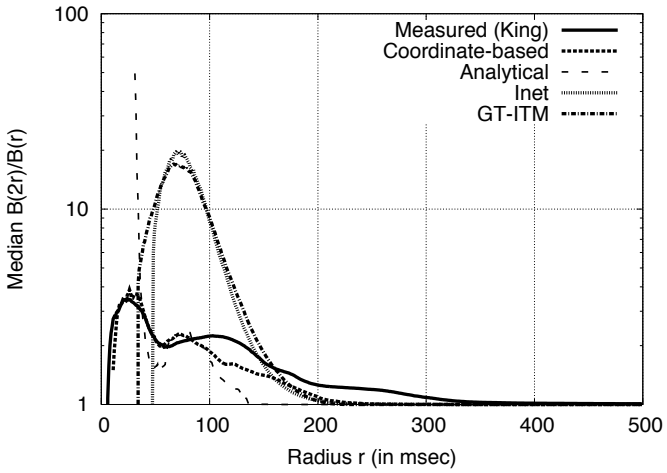

Fig. 19.7: Simulation results for spatial growth of the modelled delay spaces.

even though there is also a noticeable difference in the results obtained by former two delay spaces.

19.7 Summary

Simulation is probably the most important tool for the validation and performance evaluation of P2P systems. However, the obtained simulation results may strongly depend on a realistic Internet model. Several different models for the simulation of link delays have been proposed in the past. Most approaches do not incorporate the properties of the geographic region of the host. Hosts in a generated topology thus have overly uniform delay properties. The analytical approach, on the other hand, does not provide a jitter model that reflects the different regions and the absolute delays differ from more realistic approaches. Both the King model and our proposed coordinate-based system incorporating data from real-world measurements yield similar results. The only major drawback of King is its limited scalability. It requires memory proportional to n^2 and available datasets are currently limited to 3997 measured hosts. Statistical scaling of this data allows to preserve delay properties, but produces solely static delay values [513].

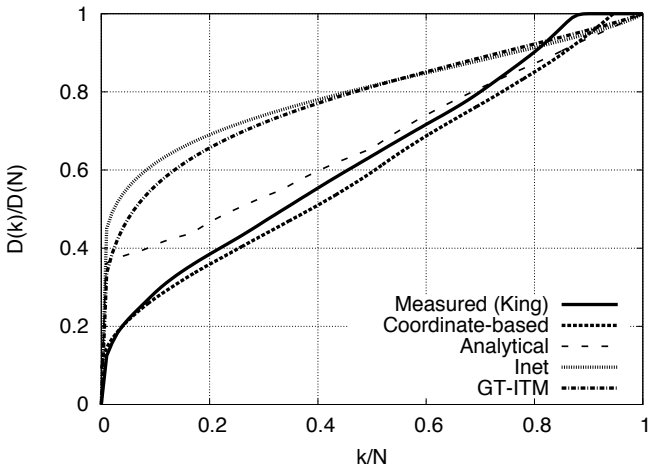

Fig. 19.8: Simulation results for the $D(k)$-function as proximity metric.

The model presented in this chapter has only linear memory costs and provides a much larger dataset of several hundred thousand hosts. Compared to topology generators the delay computation time is low. In summary, coordinate-based delay models seem to be an optimal tradeoff between many conflicting properties.