Turbo King: Framework for Large-Scale Internet Delay Measurements

Derek Leonard and Dmitri Loguinov*
Department of Computer Science
Texas A&M University, College Station, TX 77843
Email: {dleonard,dmitri}@cs.tamu.edu

Abstract—**Distance estimation and topological proximity in the Internet have recently emerged as important problems for many distributed applications [1], [10], [11], [19], [29], [31], [40], [41], [44]. Besides deploying tracers and using virtual coordinates, distance is often estimated using end-to-end methods such as King [13] that rely on the existing DNS infrastructure. However, the question of accuracy in such end-to-end estimation and its ability to produce a large-scale map of Internet delays has never been examined. We undertake this task below and show that King suffers from non-negligible error when DNS zones employ geographically diverse authoritative servers or utilize forwarders, both of which are very common in the existing Internet. We also show that King requires insertion of numerous unwanted DNS records in caches of remote servers (which is called cache pollution) and requires large traffic overhead when deployed in large-scale. To overcome these limitations, we propose a new framework we call Turbo King (T-King) that obtains end-to-end delay samples without bias in the presence of distant authoritative servers and forwarders, while consuming half the bandwidth needed by King and reducing the impact of cache pollution by several orders of magnitude. We finish the paper by evaluating Turbo King in several experiments.**

I. Introduction

Widespread interest in distance estimation in the Internet has recently evolved into a large field [4], [6], [7], [10], [11], [13], [14], [19], [21], [23], [30], [31], [32], [35], [40], [41], [43], [44], [48], [50]. The purpose of this research is to estimate or measure the latency between hosts, which can then be leveraged to provide better service to end-users and construct more efficient networks. Examples include increasing the responsiveness of online games, efficiently locating the closest server in a content distribution network, and building topologically-aware P2P networks. While the existing approaches are promising, obtaining a large-scale[1] Internet distance map for verification of virtual-coordinate approaches [7], [10], [14], [15], [19], [23], [31], [40], [41], [44] and actual use in deployed applications has proven to be a difficult task. The aim of this paper is to introduce a first step in this direction and propose a framework that allows such a service to be transparently enabled in the current Internet.

Due to the difficulty of deploying tracers [1], [11], [29], [33], [42] in every possible network, we choose to build upon an existing technique called King [13] that does not require any changes to existing protocols or access to remote computers. King approximates the distance between end-hosts using the delay between DNS servers authoritative for IP addresses of the hosts in question. While generally accepted as a sound methodology for estimating delay and used in many papers [2], [3], [5], [8], [9], [10], [12], [20], [22], [34], [36], [37], [39], [43], [48], [49], King has not been analyzed for accuracy and pitfalls since the original paper [13], nor has it been involved in measurements larger than 2500×2500 nodes. Furthermore, some of the advanced techniques suggested in [13] have never been implemented and their feasibility in practice has not been assessed in the literature.

We start the paper by identifying causes of King's inaccuracy and evaluating its suitability for large-scale measurements. We first argue that King incorrectly estimates delay when the target DNS zone contains multiple nameservers that are not geographically close to each other (e.g., outside the target domain and its BGP network). We also find that King can estimate entirely wrong delays when the source DNS zone uses forwarders, which are stand-alone servers that aggregate queries from multiple domains. In such cases, King fails to detect the presence and location of forwarders, in addition to incorrectly measuring the forwarder's query-processing delay that must be subtracted from the final measurement. In regard to overhead, King utilizes a complex multi-step process (see below for the algorithm) that requires numerous queries for each delay measurement and seeding of source DNS servers with a large number of unwanted entries. As the scale of the experiment increases, cache pollution becomes a non-trivial issue.

To overcome these drawbacks, we propose a new system called Turbo King (T-King) that streamlines the process of making distance measurements using DNS, improves their accuracy, reduces overhead, and almost entirely eliminates cache pollution. The first component of T-King is a large collection of nameservers distributed throughout the Internet, from which the closest nameserver to each end-host A is selected for use in the measurement. In the current imple-

*Supported by NSF grants CCR-0306246, ANI-0312461, CNS-0434940, CNS-0519442, and CNS-0720571.

[1]The scale considered in this paper assumes building an all-to-all delay matrix between approximately $220,000$ BGP prefixes advertised in the Internet. This is in contrast to the frequently-used latency maps today [10], [24], [31], [50] that rely on $100 - 400$ nodes in PlanetLab or $1700 - 2500$ nodes in the DNS tree.

mentation, we use periodic crawls of the DNS tree to find nameservers that can be used in the measurement and maintain this information in our server. The second component of T-King is a new measurement algorithm based on several improvements we have made to the advanced techniques in [13] that mitigate problems caused by forwarders and zones with multiple authoritative nameservers. Our approach not only reduces the number of queries and bandwidth overhead of King by more than 50%, but also achieves higher accuracy and a factor of N reduction in the number of polluted cache entries at each remote server for an $N \times N$ latency measurement.

We finish the paper by showing how to build the current database of DNS servers in Turbo King, examining how likely King is to experience its drawbacks in practice, and assessing the effect of these drawbacks on King's delay estimation. We first perform a reverse DNS crawl to discover a set of $216,843$ nameservers, out of which we find $117,817$ to be recursive and accepting queries from outside networks.[2] These servers reside in 174 countries, cover over $31,000$ BGP prefixes, and are responsible for approximately 50% of IP addresses (i.e., 828 million) advertised in BGP [38]. Further analyzing the data, we find that 33% of reverse DNS zones utilize a nameserver that neither belongs to the same BGP prefix nor the same domain as the other servers. Additionally, over 32% of recursive servers found in this study use a hidden forwarder, which suggests that a large fraction of King's measurements may be affected by the drawbacks identified in this work. We finish the paper by quantifying the effect of this bias using a small 50×50 delay matrix and comparing the estimates of King to those of Turbo King. Our results show that 15% of the measurements are different by more than 10% and 8% by more than 20%, which suggests that the magnitude of bias in King is generally mild, but nevertheless non-negligible.

The rest of the paper is organized as follows. Section II studies previous work. Sections III and IV outline issues with King. Section V introduces T-King and Section VI evaluates our method, comparing it to King. Section VII concludes the paper.

II. Background

The Domain Name System (DNS) [27], [28] is a distributed tree-based database that allows for the resolution of domain names to various types of data, most notably IP addresses. The DNS standard [28] also provides for reverse lookup of IP addresses, which is accomplished through the `IN-ADDR.ARPA` domain tree. There are several types of servers and clients that operate on DNS and to avoid confusion we introduce the following terminology. In this paper, a *recursive resolver* is a server that queries the DNS and returns answers to end-hosts. *Nameservers* are DNS servers that maintain authoritative data about a subset (i.e., zone) of the domain space. *Recursive nameservers* act as both a recursive resolver and a nameserver simultaneously. An *open resolver* is either a recursive nameserver or a recursive resolver that responds to recursive queries for arbitrary zones from hosts *outside* its local network.

[2]Other techniques (such as those in [26]) can significantly expand this database. It is believed that there are between $580,000$ [46] and $975,000$ [25] open recursive DNS servers in the Internet.

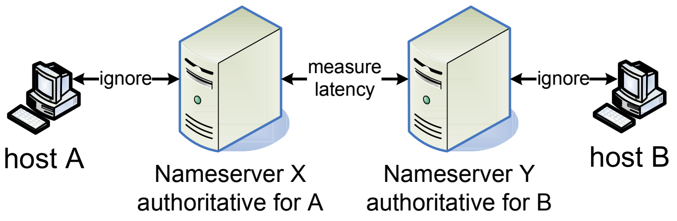

ignore
measure latency
ignore
host A
Nameserver X authoritative for A
Nameserver Y authoritative for B
host B

Fig. 1. King estimates the latency from host A to B.

King [13] uses existing DNS infrastructure to measure the latency between two hosts on the Internet. The method relies on the fact that open recursive nameservers on the Internet will attempt to resolve any valid request, which forces them to query *remote* nameservers for the proper response. The time that these queries take to be processed can be measured to determine the distance between the two nameservers. In order for the measurements to apply to *arbitrary* hosts, Gummadi *et al.* assume that end-hosts on the Internet are within close proximity to the authoritative DNS nameserver that maintains DNS information about their IP address. Given this assumption, King approximates the delay between hosts A and B using the latency between their authoritative servers X and Y as shown in Fig. 1. Heuristics are used to choose which authoritative nameserver to include in the measurements, the details of which can be found in [13].

III. Understanding Original King

We refer to the main technique proposed by Gummadi *et al.* in [13] as *Original King* (O-King). O-King has been used extensively in the literature [2], [3], [5], [8], [9], [10], [12], [20], [22], [34], [36], [37], [39], [43], [48], [49] as a way to easily collect latency information from the Internet; however, no formal or detailed analysis of its pitfalls exists to date. We first describe the measurement algorithm used by O-King, which is necessary for understanding its limitations and our proposed system later in the paper.

A. Measurement Algorithm

We start by defining terminology. Throughout the rest of the paper, a *query* is defined as a single DNS request sent to a remote server and an *answer* is the response to a query. Queries are either recursive or iterative as defined by the DNS specification [27]. Given time t_s when a query is sent and t_r when the answer is received for that query, we define a sample s to be $t_r - t_s$. We are now ready to detail the O-King algorithm that measures delay between two nameservers.

The O-King process is illustrated in Fig. 2, where `ns.example.com` is a recursive nameserver chosen by O-King as "close" to the desired IP. In the figure, each query is labeled as either `RQ` for recursive query or `IQ` for iterative query. Answers are labeled with `A`. A *seed* recursive query, which is represented by message numbers 1–4, is sent to

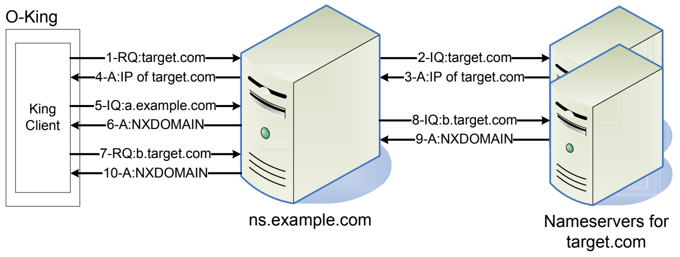


Fig. 2. O-King query sequence.

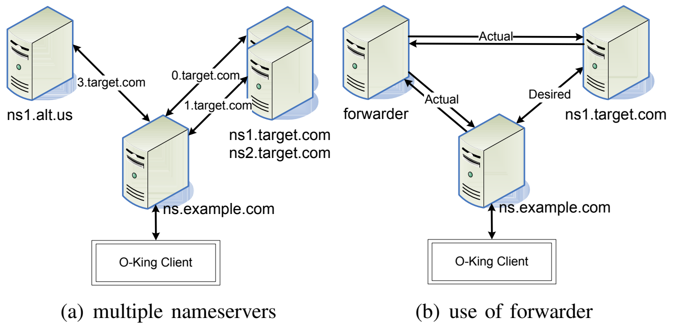


Fig. 3. O-King query sequence for two different server configurations.

ns.example.com for the target.com domain to ensure direct contact between the two for subsequent measurements. Messages 5 and 6 show the *local latency sample* L_i between the O-King client and ns.example.com, which is accomplished by a simple iterative query that can be repeated to improve accuracy. Illustrated by messages 7–10 is the *remote latency sample* R_i, which uses a recursive query to measure the delay from the O-King client to target.com (via ns.example.com) and also can be repeated. The resulting latency estimate between ns.example.com and target.com produced by O-King is $\min\{R_i\} - \min\{L_i\}$. One of the features that makes O-King so attractive is its ease of use; however, it has certain drawbacks that we discuss in the remainder of this section.

B. Zones with Multiple Authoritative Nameservers

In the original specification for DNS [27], [28], it is recommended that authoritative nameservers be placed in geographically diverse locations on separate networks. Thus, if connectivity is interrupted at one of the sites, the remaining nameservers would maintain availability for the zone. Queries for a particular zone are sent to one address in the group of nameservers, but the decision about *which* nameserver to query is left up to the individual resolver implementation. As O-King requires at least four [13] samples to converge to an accurate measurement, *different* nameservers are potentially used for each sample. While this is of little consequence if all nameservers for a zone are on the same network, in cases where the DNS specification is strictly followed the samples could be very different, leading to inaccuracy in the final latency estimation. This issue is illustrated in Fig. 3(a) for three samples taken by the O-King client, where the authoritative nameservers for the target.com zone are ns1.target.com, ns2.target.com, and ns1.alt.us.

C. DNS Forwarders

Another potential issue for O-King measurements is the use of forwarders on the Internet by system administrators. A forwarder serves as an aggregation point for DNS queries initiated from within a network that target external destinations. If a recursive nameserver that is configured to forward messages receives a recursive query for a zone it has no authority over, it sends the query to the forwarder *without notifying* the end-user. The forwarder then resolves the query instead of the recursive nameserver. This process is illustrated in Fig. 3(b), where direct contact is intended between ns.example.com and ns1.target.com, but the query is routed through the forwarder instead. The presence of forwarders is undetectable by O-King and compromises the assumption that there is direct contact between ns.example.com and ns1.target.com, leading to an invalid latency estimate.

D. Cache Pollution

The final concern that arises from the use of O-King is the impact it has on the nameservers used for latency estimates. While the purpose of an authoritative DNS nameserver is to provide accurate information about the data under its control to the global Internet, the purpose of a DNS cache is to reduce latency strictly for *local* users, those end-hosts that principally rely on the nameserver to resolve queries on their behalf. Given that DNS caches are intended to benefit these users, we define *cache pollution* to be the insertion of DNS zone data that has not been requested by a local user into the cache of a nameserver.

O-King uses a seed query to force the recursive nameserver to cache the NS (nameserver) and A (IP address) records of *all* target authoritative nameservers. While this is unlikely to cause performance problems on a small scale, initiating billions of O-King queries could lead to a large proportion of the cache containing information that was not requested by local users. Furthermore, local administrators are unlikely to view this intrusion as benign and may take preventative steps, jeopardizing future measurements using O-King.

IV. Understanding Direct King

We refer to the second technique proposed in [13] as *Direct King* (D-King), which involves a modification of the O-King measurement algorithm that allows for specification of a single nameserver from the target zone. While not mentioned explicitly in the original paper, all other aspects of D-King (i.e., nameserver selection, end-to-end estimation assumptions) we assume to be equivalent to O-King. To our knowledge, only Ballani *et al.* [3] have partially implemented D-King, which was required for their study of IP Anycast as deployed by DNS root servers. There was no study or analysis of D-King in [13]. We start by describing the D-King algorithm and later discuss some of its drawbacks.

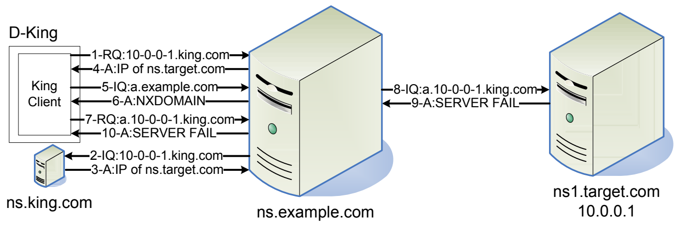


Fig. 4. D-King query sequence.

A. Measurement Algorithm

The D-King latency estimation process is illustrated in Fig. 4, where `ns.example.com` is again a recursive nameserver. In contrast to O-King, where the query is sent to one or more nameservers responsible for the `target.com` domain, D-King allows the user to pick a single authoritative nameserver, which in this case is `ns1.target.com`. To accomplish this, D-King requires that a domain name be registered and a nameserver set up to resolve queries for said domain. In the figure, `king.com` is the example domain and `ns.king.com` is its authoritative nameserver.

D-King first requires a seed query to guarantee direct contact between `ns.example.com` and `ns1.target.com`, which is illustrated in the figure by messages 1–4. To do this, D-King initiates a recursive query to `ns.example.com` for `10-0-0-1.king.com`, which encodes the IP address of `ns1.target.com` into the query. As `ns.king.com` is authoritative for the `king.com` domain, it receives the query and uses the encoded address to respond that `ns1.target.com` is authoritative for the query, which is then *cached* at `ns.example.com`. By doing so, `ns.example.com` will now automatically forward queries for `10-0-0-1.king.com` directly to `ns1.target.com`. Once the cache is seeded, the actual latency measurements can be taken. Local sample L_i, represented as messages 5 and 6 in the figure, is taken in the same fashion as O-King. Remote sample R_i, illustrated by messages 7–10, is recorded by sending queries for random subdomains of `10-0-0-1.king.com` to `ns.example.com`, which directly queries `ns1.target.com` as a result. Since `ns1.target.com` is not actually authoritative for the zone, it responds with an error indication, which is then echoed back to the D-King client. The final latency estimate produced by D-King is calculated in the same manner as that in O-King.

B. Additional Complexity

While D-King indeed eliminates the issue of zones with multiple authoritative nameservers affecting the latency estimate, the cost of this improvement is that the D-King client must explicitly specify the target nameserver, which is not required by O-King. It is not mentioned in [13] exactly how this should be accomplished, but the same heuristic approach for discovering a close recursive nameserver applies in this case as well. Furthermore, a domain must be registered and a nameserver set up to respond to queries in the way D-King requires. One of the major benefits of O-King is that latency estimates can be obtained from any machine with an Internet connection, whereas D-King requires this extra infrastructure. The individual must decide whether the additional complexity is worth the improved accuracy.

C. DNS Forwarders

Along with O-King, the use of forwarders on the Internet affects D-King latency estimates as well. The D-King client and authoritative nameserver for the measurement (e.g., `ns.king.com` in Fig. 3(b)) are separate entities that only communicate through the query encoded with the IP address of the target nameserver. It is inconsequential to `ns.king.com` that a *different* nameserver (i.e., the forwarder) than the one intended by the D-King client sends it the query and caches the response. Because of this lack of communication between the components of the D-King latency estimates, forwarders remain undetected and affect the results in the same manner as discussed in the O-King case.

D. Cache Pollution

The seed query required by D-King plants authoritative data for the registered domain (e.g., `king.com`) at the recursive nameserver in a similar fashion to that required by O-King. However, there are differences in the impact on local DNS caches. The O-King seed query forces the caching of data for *all* authoritative nameservers of the target zone, whereas D-King caches data for a single nameserver. In contrast to O-King, where the cached entries might have some future use to the local users, the D-King entry is *only* useful to the latency estimate. At the scale of billions of queries, if D-King is used the nameserver's cache would contain fewer entries than O-King, but those entries would be entirely useless to local users.

V. TURBO KING

In this section we propose Turbo King (T-King) to address the drawbacks previously highlighted. We start by giving a high-level overview of the system then finish the section with detailed descriptions of the various components.

A. Design

Turbo King is a stand-alone service that accepts as arguments the IP addresses of end-hosts A and B from the Internet and returns the estimated latency from host A to B. It is currently implemented to resolve single estimate requests for end-host pairs. Further information on the deployment of T-King can be found in [45].

To accomplish this goal, Turbo King maintains a large list S of N nameservers positioned throughout the Internet, which includes both recursive nameservers and non-recursive authoritative nameservers found in the DNS hierarchy. This list allows us to discover the closest nameserver without relying strictly on heuristic methods or assuming that the authoritative nameserver responsible for A's IP address is the closest nameserver to A. Turbo King first uses BGP data [38] to match the IP address of A to a recursive nameserver. If a match is found, the two are likely to reside in the same network. If no matching nameserver is found in the same network as the end-host, we simply find the recursive nameserver that has the longest matching prefix to A or select the default nameserver authoritative for A's IP address. Turbo King then repeats the same process for B but expands the set of possible nameservers to include those that are not recursive. This is done because the target nameserver need not resolve recursive queries for the estimate to succeed.

Given the two nameservers, Turbo King then generates a latency estimate between them (the algorithm is described below) and returns the result. T-King operates in one of two modes. The default is *passive*, whereby T-King waits for requests before generating latency estimates. Estimates are cached for a configurable amount of time (e.g., 30 minutes) such that subsequent requests using the same two nameservers do not trigger a new measurement. This mode puts the least strain on resources as it only visits popular destinations. The optional mode we call *active*, in which Turbo King preemptively takes latency estimates between nameservers on the list so as to eventually obtain an entire $N \times N$ delay matrix.[3] This mode consumes more resources and possibly produces estimates that are never used, but it reduces the user-perceived delay and allows the matrix to be directly downloaded for use in applications and other research studies.

B. Discovering Nameservers

Turbo King is most effective when its list S of nameservers is large, such that at least one nameserver is "close" to every IP address that is currently in use on the Internet. The current version of T-King compiles its list of nameservers by performing exhaustive crawls[4] of the `IN-ADDR.ARPA` reverse DNS tree using the techniques introduced in [18]. In contrast to [18], which accepts *cached* (i.e., non-authoritative) entries to queries because they are only interested in the number of hosts represented in the tree, our crawler probes the entire depth of the reverse tree by accepting *only* authoritative answers, which maximizes the number of nameservers found. Results from this crawl are presented in the next section, but we should note that significantly larger datasets can be built using other techniques (e.g., $N = 333,963$ in [26] and over $580,000$ using port-53 scanning [46]). We are considering the intrusiveness and overhead of these approaches for later augmenting set S.

C. Measurement Algorithm

Our proposed algorithm is illustrated in Fig. 5, where Turbo King operates as a multi-threaded application with both the client and server operations communicating seamlessly. This allows timestamps to be taken for every packet sent or received by our software, which reduces the number of queries required to complete an estimate. During step

[3]For $N = 117,863$ used in the current version and one query per 22 seconds per DNS server, the entire matrix consisting of 13.8 billion measurements can be built in 30 days.

[4]Analysis shows that 85% of nameservers found by T-King in Nov. 2006 were active in Dec. 2007, which suggests that monthly or even annual re-scanning of the tree should keep the DNS server set relatively fresh.

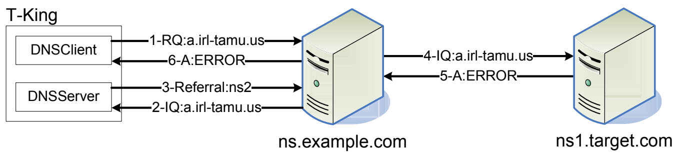


Fig. 5. Turbo King query sequence.

1, `DNSClient` takes a timestamp and initiates a query to `ns.example.com` for the domain we control, namely `irl-tamu.us`. Since `DNSServer` is listed with the `.us` registrar as the authoritative nameserver for this domain, host `ns.example.com` recursively queries `DNSServer` in step 2. At step 3, our software takes another timestamp and answers with a referral saying that `ns1.target.com` is authoritative for the query, but sets the TTL for this information to zero, meaning that it should *not* be cached [28].[5] Nameserver `ns.example.com` then directly queries `ns1.target.com`, which answers with some form of error indication (steps 4–5). That error indication is forwarded back to `DNSClient` in step 6, which then takes the third and final timestamp, allowing us to estimate the latency between nameservers `ns.example.com` and `ns1.target.com` as $d_{36} - d_{12}$ where d_{ij} is the delay between steps i and j. Thus, Turbo King is able to determine the latency between `ns.example.com` and `ns1.target.com` without seeding the cache of `ns.example.com` by judiciously taking timestamps at every point of communication between `ns.example.com` and the T-King software.

D. Detection and Avoidance of Forwarders

While both O-King and D-King are unable to detect forwarders, they are simple to detect with Turbo King due to its integrated infrastructure and can be eliminated from the measurement. Because T-King acts as both the client *and* the server application for the latency estimate, it simply compares the IP addresses that are used to contact `DNSClient` and `DNSServer` respectively for a particular query. If different IP addresses are used, T-King excludes the original IP from the list of recursive nameservers and determines if the forwarder allows for recursion, adding it to the list of possible nameservers if so. A new closest server to the IP is retrieved from the list of recursive nameservers and the latency estimate restarts. While there is some small additional delay in returning an answer to the end-user when in passive mode, the resulting estimate is not tainted by the presence of a forwarder.

VI. Evaluation

In this section we evaluate the effectiveness of Turbo King for providing accurate latency measurements and its suitability for large-scale studies compared to O-King and D-King. We start by discussing our efforts to discover a large number of nameservers, then perform several real-world measurements to compare the three algorithms.

[5]We found that 35 of the $117,817$ discovered recursive nameservers were either misconfigured or non-compliant with [28] and ignored zero TTL.

TABLE I
RESULTS OF REVERSE DNS CRAWL (3.8 GHz PENTIUM 4)

	T-King	ISC [18]
Month run	Nov. 2006	Jul. 2006
Duration (hours)	33.8	240
Queries/Sec	$5,300$ (2.3 mb/s)	751 (0.3 mb/s)
Queries Completed	$649,270,000$	N/A
IPs Discovered	$439,431,355$	$439,286,364$
Nameservers	$216,843$	$89,592$
Recursive Nameservers	$117,817$	N/A

TABLE II
COVERAGE OF THE INTERNET WITH DISCOVERED SERVERS

	All	Recursive	Total
Countries	190	174	232 [17]
AS	$13,017$	$10,895$	$23,773$ [16]
BGP Prefixes	$48,196$	$31,059$	$219,110$ [38]
IPs covered	$1,031,736,562$	$828,675,500$	$1,642,441,178$
Web servers	$3,192,918$	$2,659,379$	$3,638,433$
Gnutella peers	$1,734,483$	$1,338,217$	$3,534,300$

A. *Results from Reverse DNS Crawl*

Because of the large number of queries required to complete the `IN-ADDR.ARPA` crawl, we designed and implemented a multi-threaded DNS resolver to collect a list of nameservers and authority data for *all* zones in the reverse lookup tree. The results of one particular crawl executed in November 2006 are summarized in Table I, where both Turbo King and ISC [18] found roughly the same number of IP addresses in the tree; however, our crawler was approximately seven times faster and discovered 2.4 times more nameservers. Examination of the nameservers we discovered revealed that $117,817$ of them support recursive queries. Using the T-King client, we profiled each of the recursive nameservers in our list and found that 32% use a forwarder to resolve queries for zones not under their control.

We next study the coverage of the Internet by all discovered nameservers and the subset of nameservers that are recursive, which is illustrated in Table II. This data shows that Turbo King contains a nameserver in 190 countries, covering over 13 thousand ASes, 48 thousand BGP prefixes [38], and 1.03 billion IP addresses out of 1.6 billion advertised by BGP [38]. We performed further analysis of how well the discovered BGP prefixes cover 3.5 million Gnutella peers found in prior work [47] and 3.6 million web servers (hosting over 6.3 billion webpages) found by our unrelated web-crawling project. These numbers show that 49% of peers and 88% of web servers reside in BGP prefixes that contain at least one nameserver discovered by T-King.

For the subset of nameservers that are recursive, Turbo King found nameservers in 174 countries, representing nearly 11 thousand ASes, 31 thousand BGP prefixes, and 828 million IP addresses. This resulted in a coverage of 37% of Gnutella peers and 73% of webservers. While T-King is able to find a nameserver in the same network for a large percentage of end-hosts (especially web servers), the relatively low percentage of Gnutella peers covered indicates that we should aim to discover more recursive resolvers used by home-based Internet connections in the future.

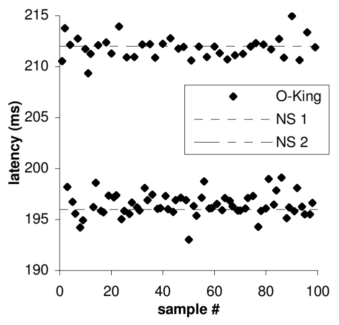
1) Analyzing zone authority data: Of further interest is the percentage of zones in the reverse lookup tree that contain multiple authoritative nameservers, which we examined by recording the set of nameservers authoritative for every zone during the `IN-ADDR.ARPA` crawl. The accuracy of O-King estimates is only significantly affected if one or more of the nameservers for a zone is in a *different* network, making it likely for O-King to produce conflicting results over multiple samples. We downloaded $219,110$ BGP prefixes from Route-Views [38] and matched each nameserver's IP to one or more prefixes, then examined the nameserver set for every zone in the reverse lookup tree. We found that 49% of reverse lookup zones contain at least one nameserver in their set that is in a different network. While this is a striking result, it is possible that the unmatched nameserver could be in another network under the same administrative control that is well-connected to the rest of the nameservers in the set.

Accurately determining administrative control for a large number of networks is difficult, but it stands to reason that if all nameservers for a zone share a single domain name, they are more likely to be under one organization's administrative control. While the process is easy for generic top-level domains (gTLDs), it is significantly more complex for country-coded TLDs (ccTLDs) as most countries created sub-domains from which people could purchase their own domains (e.g., `.com.es`). We compiled a comprehensive list of these sub-domains for each ccTLD and hereby refer to this list and the set of gTLDs as pay-level domains (PLDs). We again evaluated the nameserver set for each zone and found that 33% have at least one nameserver in their authority set that both resides in a different network and has a different PLD than the other nameservers. It is very likely that the accuracy of O-King queries for these zones will be negatively impacted.

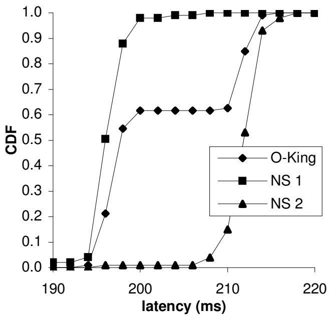
B. *Causes of Inaccuracy*

In this section we compare O-King and D-King to Turbo King using latency estimates they produce from the Internet. To remove variability caused by differences in architecture, we implemented all three algorithms using the same timing and socket mechanisms and ran all of the tests from a single Windows 2003 x64 machine. To highlight the differences in accuracy, we focus only on the actual latency estimate between nameservers and note that T-King should perform *no worse* than O-King or D-King in selecting a "close" recursive nameserver. In many cases it will perform better, but we leave such analysis for future work.

1) Zones with Multiple Authoritative Nameservers: To illustrate the impact of authoritative nameservers in different networks on O-King estimates, we chose a zone with two authoritative nameservers and used O-King to generate 100 latency estimates to a target IP in the zone. We then used D-King to estimate the latency to the two individual authoritative nameservers for the zone. In Fig. 6(a) the sequence of O-King estimates are individual points and the two D-King estimates

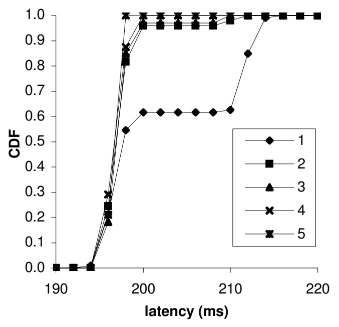


(a) sample sequence

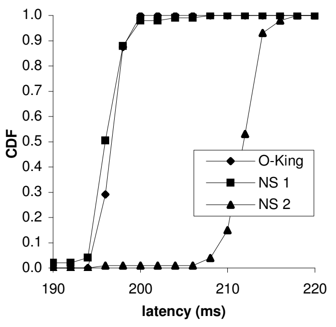


(b) CDF of latencies

Fig. 6. Comparison of O-King to D-King for zone with two nameservers.

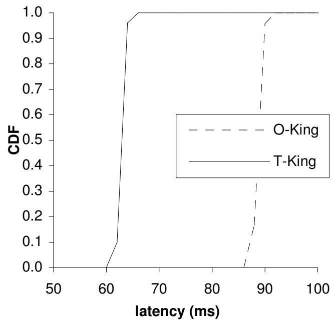


(a) latency estimate per sample size

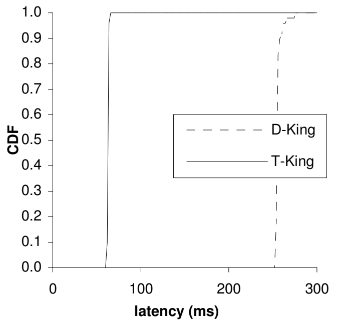


(b) four-sample O-King vs. D-King

Fig. 7. Convergence of O-King estimate for zone with two nameservers.



(a) O-King with forwarder



(b) D-King with forwarder

Fig. 8. Forwarder affect on O-King and D-King (single nameserver zone).

are represented as lines. O-King performs as expected and vacillates between the two nameservers arbitrarily. This is further demonstrated in Fig. 6(b), where roughly 60% of the time O-King chose `NS 1` as the preferred server. We confirmed that this behavior also occurs in zones with more than two nameservers, but omit these examples for brevity.

We next analyze the convergence properties of O-King measurements for zones with multiple authoritative nameservers. To determine exactly what happens to the latency estimate when the number of samples increases, we use the measurement data from above and plot in Fig. 7(a) the CDF of latency estimates for sample sizes one through five. From inspecting the figure, it quickly becomes apparent that the higher latency samples are ignored and are effectively removed from the overall estimate as the sample size increases. This is further illustrated in Fig. 7(b), where the latency estimated by O-King using four samples is plotted with D-King measurements using one sample. As the figure clearly shows, the O-King measurement is nearly identical to the measurement given by D-King to `NS 1`.

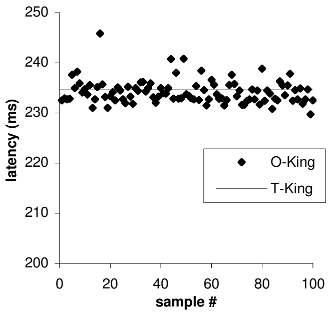
There are two insights that can be gained from this behavior. The first is that the requirement of at least four samples per measurement proposed in [13] for O-King is at least partially due to the natural differences in latency between multiple authoritative nameservers for a particular zone. In contrast, D-King provides the same latency estimate with one sample in this case. The second issue is that O-King always biases its estimates towards the *lowest* latency nameserver of a zone. While in some cases this might be the server located closest to the target end-host, there is no evidence that this happens in the general case.

2) DNS Forwarders: The impact of forwarders on both O-King and D-King latency estimates largely depends on the proximity of the forwarder to the original recursive nameserver. If the two servers are on the same local network, any additional latency should be rather small. To quantify the likelihood of this event, we matched both the forwarder and the original recursive nameserver to their advertised BGP prefixes from RouteViews [38] and discovered that 45% of the time the two servers did not reside on the same network. To demonstrate the inaccuracy introduced by the presence of forwarders, we took 100 latency estimates using all three algorithms from a recursive nameserver known to use a forwarder to a zone with a single authoritative nameserver (this rules out effects from multiple authoritative nameservers on O-King). The resulting latency estimates are illustrated in Fig. 8(a), which compares O-King to T-King, and in Fig. 8(b), which compares D-King to T-King. In both figures O-King and D-King overestimate latency due to the presence of the forwarder, whereas T-King does not. The D-King estimate is larger than O-King due to multiple attempts to resolve the query by the forwarder, a problem that is mentioned in [13] and accounted for in T-King.

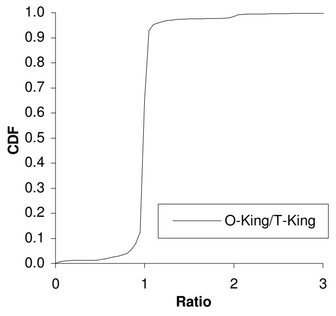
C. Measurement-based Comparison

To study Turbo King in more depth, we performed $2,450$ latency estimates on the Internet using 50 recursive nameservers from the previously discovered set for both T-King and O-King over various measurement sample sizes. From this data we show that T-King measurements are indeed different from those produced by O-King. We then show that Turbo King converges to a consistent latency estimate in two samples instead of the four suggested in [13]. D-King is omitted from this section due to space constraints, but the results are consistent with those found in the previous section. D-King is more accurate than O-King, but less so than T-King due to its inability to detect forwarders.

1) Turbo King versus O-King estimates: Before comparing Turbo King to O-King in a general case, we first consider the two algorithms for a target zone with a *single* authoritative

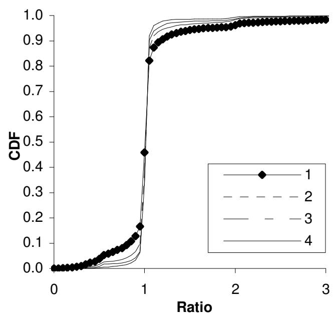


(a) single nameserver target zone

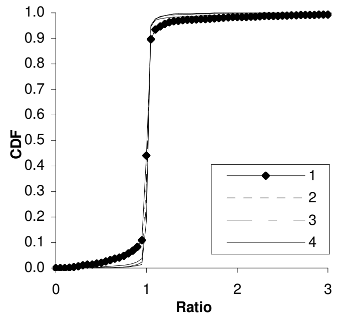


(b) ratio of O-King/T-King

Fig. 9. T-King vs. O-King to validate implementation and show differences.



(a) O-King/O-King by sample size



(b) T-King/T-King by sample size

Fig. 10. Convergence of measured latencies for O-King and T-King.

nameserver using a recursive nameserver that we verified does not use a forwarder. Because we eliminated all of the factors that skew O-King estimates, the two should produce the same value. The result is illustrated in Fig. 9(a), with the estimates produced by O-King as data points and the average estimate by T-King as a line to allow the reader to distinguish between the two. The figure clearly shows that T-King produces latency estimates that are equivalent to O-King in such idealized cases.

We next compare the $2,450$ latency estimates produced by Turbo King to those by O-King, which is shown in Fig. 9(b). To highlight differences between the two, we generated a ratio of the estimates by dividing O-King's value by T-King's, so that if O-King and T-King produced identical values, the CDF would be a straight line at one on the x-axis. Note that in this case we used four samples for each estimate as suggested in [13]. From the data, 15% of O-King estimates are more than 10% different from T-King measurements, and 8% of O-King measurements are more than 20% different from those generated by T-King.

2) Convergence of Estimates: Previously, we showed that zones with multiple authoritative nameservers are one of the reasons O-King requires at least four samples to produce a latency estimate. In this section, we expand that study by examining the convergence properties of both T-King and O-King, showing that over a wide range of estimates Turbo King converges to a consistent estimate with fewer samples than required by O-King. To accomplish this, we repeated the above $2,450$ latency estimates using sample sizes varying from one to four for both algorithms. We collected two estimates for each sample size and calculated the ratio of both O-King to O-King and T-King to T-King. The goal is to provide *consistent* estimates for latency, so we plotted the CDF of the ratio in Fig. 10(a) for O-King and Fig. 10(b) for T-King, with each line representing the number of samples used to produce the estimate. In the O-King case, illustrated in Fig. 10(a), improvement in the consistency of estimates is apparent as the number of samples increases to four, whereas in the Turbo King case (Fig. 10(b)) the greatest improvement is from one sample to two, with little afterward. From these graphs we conclude that the recommendation of four samples in [13] is sound for O-King and that T-King produces an accurate sample using only two samples.

D. Overhead Analysis

In this section we study the resources required of DNS nameservers and the Internet for all three algorithms. In particular, we are interested in how the three compare for large-scale estimates involving more than $100,000$ recursive nameservers discovered by Turbo King. We start by examining the number of queries sent to capture the network overhead, then discuss cache pollution for large-scale measurements.

1) Network Overhead: To study network overhead, we consider the number of queries required to perform all-to-all latency estimates for the $100,000$ recursive nameservers, which is 10 billion estimates. In this calculation, we included every query initiated either by or on behalf of the measurement client, and used the number of samples required to produce consistent latency estimates: four for O-King and two for D-King and T-King. Due to the lack of seeding, Turbo King requires 70 billion queries to complete 10 billion latency estimates. D-King needs 100 billion queries for the measurement, which is 1.43 times more than required by T-King. Finally, O-King uses 150 billion queries, or 2.14 times more than Turbo King and 1.5 times more than D-King. Thus, designing T-King to be more accurate and to avoid seeding led to a significant reduction in bandwidth usage.

We next consider the impact each algorithm has on DNS caches under the same measurement conditions.

2) Cache Pollution: We examine cache pollution by calculating the total number of DNS records inserted into the cache of the $100,000$ recursive nameservers. Each entry includes two records, an `NS` record and an `A` (IP address) record. Since O-King causes recursive nameservers to seed the cache with *every* authoritative nameserver for a zone, we used the reverse crawl data to find an average of 2.4 nameservers per zone. Thus, O-King would cause the insertion of 48 billion entries into cache for the nameservers used in the measurement. D-King needs a single set of records for each latency estimate, which means that 20 billion entries would be saved in caches on its behalf. Turbo King only requires that the local domain (e.g., `irl-tamu.us`) be cached at each recursive nameserver, which implies merely $200,000$ total cache pollution entries. To compare, Turbo King requires 0.0004% of the total entries caused by O-King and 0.001% of those initiated by D-King, clearly making Turbo King much more appropriate for large-scale measurement studies.

VII. Conclusion

In this paper we proposed the Turbo King latency estimation framework and showed that it was more accurate than prior methods while at the same time requiring fewer samples to produce an accurate latency measurement and scaling significantly better in terms of overhead and cache pollution. More information on the deployment of Turbo King can be found in [45].

Future work includes running T-King in active mode on the Internet, locating more nameservers to increase coverage for home-based end-users, verifying existing distance estimation techniques, and creating a system that leverages both theoretical approaches and actual latency estimates on the Internet.

References

[1] A. Adams, J. Mahdavi, M. Mathis, and V. Paxson, "Creating a Scalable Architecture for Internet Measurement," in *Proc. ISOC INET*, Jul. 1998.

[2] A. Baggio and M. van Steen, "Distributed Redirection for the World-Wide Web," *Computer Networks*, vol. 49, no. 6, pp. 743–765, Dec. 2005.

[3] H. Ballani, P. Francis, and S. Ratnasamy, "A Measurement-based Deployment Proposal for IP Anycast," in *Proc. ACM IMC*, Oct. 2006, pp. 231–244.

[4] S. Banerjee, C. Kommareddy, and B. Bhattacharjee, "Scalable Peer Finding on the Internet," in *Proc. IEEE GLOBECOM*, Nov. 2002, pp. 2205–2209.

[5] C. Chambers, W. Feng, and W. Feng, "A Geographic Redirection Service for On-line Games," in *Proc. ACM Multimedia*, Nov. 2003, pp. 227–230.

[6] Y. Chen, K. H. Lim, R. H. Katz, and C. Overton, "On the Stability of Network Distance Estimation," *SIGMETRICS Performance Evaluation Review*, vol. 30, no. 2, pp. 21–30, Sep. 2002.

[7] M. Costa, M. Castro, A. Rowstron, and P. Key, "PIC: Practical Internet Coordinates for Distance Estimation," in *Proc. IEEE ICDCS*, Mar. 2004, pp. 178–187.

[8] R. Cox, F. Dabek, F. Kaashoek, J. Li, and R. Morris, "Practical, Distributed Network Coordinates," *ACM SIGCOMM Comp. Comm. Rev.*, vol. 34, no. 1, pp. 113–118, Jan. 2004.

[9] F. Dabek, J. Li, E. Sit, J. Robertson, M. Kaashoek, and R. Morris, "Designing a DHT for low latency and high throughput," in *Proc. USENIX NSDI*, Mar. 2004, pp. 85–98.

[10] F. Dabek, R. Cox, F. Kaashoek, and R. Morris, "Vivaldi: A Decentralized Network Coordinate System," in *Proc. ACM SIGCOMM*, Aug. 2004, pp. 15–26.

[11] P. Francis, S. Jamin, C. Jin, Y. Jin, D. Raz, Y. Shavitt, and L. Zhang, "IDMAPS: A Global Internet Host Distance Estimation Service," *IEEE/ACM Trans. Networking*, vol. 9, no. 5, pp. 525–540, Oct. 2001.

[12] L. Garcés-Erice, K. Ross, E. Biersack, P. Felber, and G. Urvoy-Keller, "Topology-Centric Look-Up Service," in *Proc. NGC*, Sep. 2003, pp. 58–69.

[13] K. P. Gummadi, S. Saroiu, and S. D. Gribble, "King: Estimating Latency between Arbitrary Internet End Hosts," in *Proc. ACM IMW*, Nov. 2002, pp. 5–18.

[14] J. Guyton and M. Schwartz, "Locating Nearby Copies of Replicated Internet Servers," in *Proc. ACM SIGCOMM*, Aug. 1995, pp. 288–298.

[15] S. Hotz, "Routing Information Organization to Support Scalable Inter-domain Routing with Heterogeneous Path Requirements," Ph.D. dissertation, University of Southern California, Oct. 1994.

[16] Geoff Huston. [Online]. Available: http://bgp.potaroo.net.

[17] Internet Assigned Numbers Authority (IANA). [Online]. Available: http://www.iana.org.

[18] ISC Internet Domain Survey. [Online]. Available: http://www.isc.org/index.pl?/ops/ds/.

[19] C. Kommareddy, N. Shankar, and B. Bhattacharjee, "Finding Close Friends on the Internet," in *Proc. IEEE ICNP*, Nov. 2001, pp. 301–309.

[20] J. Ledlie, P. Gardner, and M. Seltzer, "Network Coordinates in the Wild," in *Proc. USENIX NSDI*, Apr. 2007, pp. 299–311.

[21] L. Lehman and S. Lerman, "PCoord: Network Position Estimation Using Peer-to-Peer Measurements," in *Proc. IEEE NCA*, Sep. 2004, pp. 15–24.

[22] J. Li, "Routing Tradeoffs in Dynamic Peer-to-Peer Networks," Ph.D. dissertation, Massachusetts Institute of Technology, 2005.

[23] H. Lim, J. C. Hou, and C.-H. Choi, "Constructing Internet Coordinate System Based on Delay Measurement," in *Proc. ACM IMC*, Oct. 2003, pp. 129–142.

[24] E. K. Lua, T. Griffin, M. Pias, H. Zheng, and J. Crowcroft, "On the Accuracy of Embeddings for Internet Coordinate Systems," in *Proc. ACM IMC*, Oct. 2005, pp. 125–138.

[25] The Measurement Factory DNS Survey Executive Summary. [Online]. Available: http://dns.measurement-factory.com/surveys/sum1.html.

[26] The Measurement Factory DNS Surveys. [Online]. Available: http://dns.measurement-factory.com/surveys/.

[27] P. Mockapetris, "Domain Names – Concepts and Facilities," *IETF RFC 1034*, Nov. 1987.

[28] P. Mockapetris, "Domain Names – Implementation and Specification," *IETF RFC 1035*, Nov. 1987.

[29] National Laboratory for Applied Network Research (NLANR). [Online]. Available: http://moat.nlanr.net/.

[30] T. S. E. Ng and H. Zhang, "A Network Positioning System for the Internet," in *Proc. USENIX*, Jun. 2004, pp. 141–154.

[31] T. S. E. Ng and H. Zhang, "Predicting Internet Network Distance with Coordinates-Based Approaches," in *Proc. IEEE INFOCOM*, Jun. 2002, pp. 170–179.

[32] M. Pias, J. Crowcroft, S. Wilbur, T. Harris, and S. Bhatti, "Lighthouses for Scalable Distributed Location," in *Proc. IPTPS*, Feb. 2003, pp. 278–291.

[33] PingER Project at Stanford. [Online]. Available: http://www-iepm.slac.stanford.edu/pinger/.

[34] M. Rabinovich, S. Triukose, Z. Wen, and L. Wang, "DipZoom: The Internet Measurements Marketplace," in *Proc. IEEE INFOCOM*, Apr. 2006.

[35] S. Ratnasamy, M. Handley, R. Karp, and S. Shenker, "Topologically-Aware Overlay Construction and Server Selection," in *Proc. IEEE INFOCOM*, Jun. 2002, pp. 1190–1199.

[36] S. Ren, L. Guo, and X. Zhang, "ASAP: An AS-Aware Peer-Relay Protocol for High Quality VoIP," in *Proc. IEEE ICDCS*, Jul. 2006, pp. 70–79.

[37] I. Rose, R. Murty, P. Pietzuch, J. Ledlie, M. Roussopoulos, and M. Welsh, "Cobra: Content-based Filtering and Aggregation of Blogs and RSS Feeds," in *Proc. USENIX NSDI*, Apr. 2007, pp. 29–42.

[38] Route-Views. [Online]. Available: http://www.routeviews.org.

[39] H. Shang and C. E. Wills, "Piggybacking Related Domain Names to Improve DNS Performance," *Computer Networks*, vol. 50, no. 11, pp. 1733–1748, Aug. 2006.

[40] Y. Shavitt and T. Tankel, "Big-Bang Simulation for Embedding Network Distances in Euclidean Space," *IEEE/ACM Trans. Netw.*, vol. 12, no. 6, pp. 993–1006, Dec. 2004.

[41] Y. Shavitt and T. Tankel, "On the Curvature of the Internet and its Usage for Overlay Construction and Distance Estimation," in *Proc. IEEE INFOCOM*, Mar. 2004, pp. 385–395.

[42] Skitter Project at CAIDA. [Online]. Available: http://www.caida.org/tools/measurement/skitter/.

[43] M. Szymaniak, G. Pierre, and M. van Steen, "Scalable Cooperative Latency Estimation," in *Proc. IEEE ICPADS*, Jul. 2004, pp. 367–376.

[44] L. Tang and M. Crovella, "Virtual Landmarks for the Internet," in *Proc. ACM IMC*, Oct. 2003, pp. 143–152.

[45] Turbo King. [Online]. Available: http://tking.irl.cs.tamu.edu/.

[46] R. Vaughn and G. Evron, "DNS Amplification Attacks," ISOTF, Tech. Rep., Mar. 2006.

[47] X. Wang, Z. Yao, and D. Loguinov, "Residual-Based Measurement of Peer and Link Lifetimes in Gnutella Networks," in *Proc. IEEE INFOCOM*, May 2007, pp. 391–399.

[48] B. Wong, A. Slivkins, and E. G. Sirer, "Meridian: A Lightweight Network Location Service without Virtual Coordinates," in *Proc. ACM SIGCOMM*, Aug. 2005, pp. 85–96.

[49] B. Zhang, T. Ng, A. Nandi, R. Riedi, P. Druschel, and G. Wang, "Measurement-Based Analysis, Modeling, and Synthesis of the Internet Delay Space," in *Proc. ACM IMC*, Oct. 2006, pp. 85–98.

[50] R. Zhang, Y. C. Hu, X. Lin, and S. Fahmy, "A Hierarchical Approach to Internet Distance Prediction," in *Proc. IEEE ICDCS*, Jul. 2006, pp. 73–81.