King: Estimating Latency between Arbitrary Internet End Hosts

Krishna P. Gummadi, Stefan Saroiu, and Steven D. Gribble
Department of Computer Science & Engineering
University of Washington, Seattle, WA, USA, 98195-2350
{gummadi,tzoompy,gribble}@cs.washington.edu

ABSTRACT

The ability to estimate network latencies between arbitrary Internet end hosts would enable new measurement studies and applications, such as investigating routing path inefficiencies on a wide-scale or constructing topologically sensitive overlay networks. In this paper we present King, a tool that accurately and quickly estimates the latency between arbitrary end hosts by using recursive DNS queries in a novel way. Compared to previous approaches, King has several advantages. Unlike IDMaps, King does not require the deployment of additional infrastructure, and unlike GNP, King does not require end hosts to agree upon a set of reference points. Unlike both IDMaps and GNP, King's estimates are based on direct online measurements rather than offline extrapolation. Because King uses existing DNS infrastructure, King scales naturally both in terms of the number of hosts that can be measured and in terms of the number of hosts performing measurements.

After describing the techniques used in King, we present an extensive evaluation and analysis of the accuracy and consistency of our tool. Specifically, our evaluation shows that the accuracy of King is significantly better than the accuracy of IDMaps, and that King tends to preserve order among its latency estimates. Finally, we describe a variety of measurement studies and applications that could benefit from our tool, and present results from one such measurement study.

Keywords—**latency measurement tool, recursive DNS**

I. INTRODUCTION

Designing a tool that can accurately, quickly, and cheaply estimate network latencies between arbitrary Internet end hosts is challenging, but such a tool could prove to be as valuable as existing network measurement tools like *ping* and *traceroute*. For example, this tool could be directly used in applications such as closest server selection, or in the construction of topology-aware multicast overlays [17]. Perhaps its greatest utility would be in drastically simplifying the execution of wide-area measurement studies at scale.

This work was supported in part by the National Science Foundation under ITR grant CCR-0121341.

Several previous measurement studies have required the ability to measure diverse collections of wide-area path latencies, including the investigation of Internet routing path inefficiencies [19], quantifying the correlation between geographical distances and IP latencies [15], measuring the efficiency of server selection in CDNs like Akamai [9], and the evaluation of *binning* strategies for overlay construction [17]. These specific studies required control over end hosts to perform their measurements, and as a result, they limited the scale of their investigation to at most one hundred end points.

In this paper, we present King, a tool that can estimate the latency between arbitrary end hosts. King is *accurate*: it is capable of generating estimates that are very close to the true path latencies, and its estimates are based on direct, online measurement. King is *easy to use*, in that no additional infrastructure needs to be deployed and the measured end hosts do not need to cooperate. King is also *fast* and *lightweight*, requiring the generation of only a few packets to produce an estimate. Similar to Sting [20] and T-BIT [14], the main insight behind King is that it is possible to use existing protocols in unanticipated ways to obtain results that were previously intractable. Specifically, King makes use of the existing DNS infrastructure in a novel manner.

Our technique relies on two observations. First, given a pair of end hosts to be measured, in most cases it is possible for King to find DNS name servers that are topologically close to the end hosts. Second, given a pair of DNS name servers, King can accurately estimate the latency between them using recursive DNS queries. Thus, King is able to use the measured latency between the name servers as an estimate of the latency between the end hosts. Although this estimate will inevitably suffer from inaccuracies, our extensive evaluation demonstrates that this estimation error is small (less than 20% error in over three-

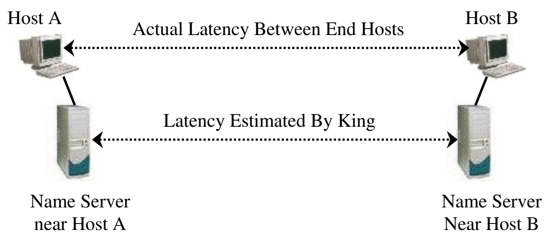


Fig. 1. King estimates the latency between two end hosts by measuring the latency between nearby DNS name servers.

quarters of generated estimates). This error is significantly smaller than that of IDMaps [6], an existing system for path latency estimation.

As King makes use of DNS, it can scale both in terms of the number of end hosts that can be measured and in terms of the number of clients that can use the tool simultaneously. Unlike GNP [12], another existing path latency estimation tool, King does not require active cooperation from end hosts.

The rest of this paper is organized as follows. In Section II, we explain in detail how King uses recursive DNS queries to estimate the latencies between end hosts, and discuss the resulting strengths and weaknesses of our approach. In Section III, we describe existing techniques to estimate Internet path latencies and compare them with King. We present a detailed, quantitative evaluation of the accuracy of our tool in Section IV, including a direct comparison with the accuracy of IDMaps. In Section V, we describe how King can be used in various applications and wide-area measurement studies, and present the results of using King to confirm a previously published result about routing path inefficiencies [19], but at a larger scale. Finally, we conclude in Section VI.

II. How King Works

King is based on two simple observations: most end hosts in the Internet are located close to their DNS name servers (see Section IV-B.2), and recursive DNS queries can be used to measure the latency between pairs of DNS servers. Thus, as shown in Figure 1, it should be possible to estimate the latency between two end hosts by locating nearby name servers and measuring the latency between them.

To measure the latency between two name servers, King issues a recursive DNS query to one name server, requesting it to resolve a name belonging to a domain for which the other server is authoritative. An example of this is shown in Figure 2: King measures the amount of time it takes to issue a recursive query to name server A for the name *xyz.foo.bar*, given that name server B is an authoritative server for the domain *foo.bar*. Name server A resolves the query by interrogating name server B, and forwarding its reply back to the client. Next, King measures the latency between the client and the name server A, either by using an ICMP ping, or perhaps by issuing an iterative[1] DNS query. By subtracting these two latencies, King produces its estimate of the latency between the two servers.

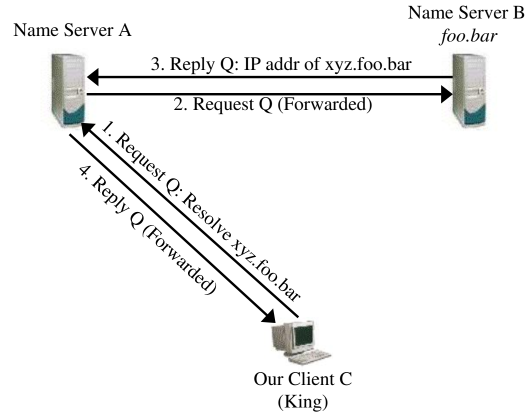


Fig. 2. The sequence of DNS messages used by King to estimate latency.

In the above discussion, we assumed that name server A directly contacts name server B instead of having to traverse the DNS hierarchy. To ensure this, we "prime" name server A with one recursive DNS query to ensure that A caches the fact that B is authoritative for *foo.bar* before King begins its measurement.

To obtain a more accurate latency estimate, King can be configured to measure the query latency multiple times; however, we cannot use the same query *xyz.foo.bar* to obtain the multiple estimates, as after the first attempt the reply would be cached by name server A. Thus, on every attempt, King issues a different query of the form *random_number.foo.bar*, where random_number is a large random number. It is very likely that this query is not a valid name in the *foo.bar* domain and in such a case, name server B would reply with a "NXDOMAIN" message.

This simplified description of the tool raises several questions. After posing four such questions, the rest of this section answers them in detail.

1. King approximates the latency between end hosts as the latency between nearby name servers. For the estimates to be accurate, we assumed that King can find name servers A and B that are close to end hosts in network topology. *Given a particular end host, how does King find such a name server?*

[1] An iterative query has the "recursion desired" bit in its header turned off; such queries are answered always by the name server from its local cache and are never forwarded.

2. We assumed that name server A would resolve the recursive query on behalf of any arbitrary host in the Internet, including our client host C. However, name server administrators could turn off recursive queries, or restrict remote clients from issuing recursive queries. *How often are such access control measures imposed in practice?*
3. We assumed that our recursive query, issued to name server A for a name in domain *foo.bar*, would be forwarded to the name server B that is picked by King. However, if there are multiple authoritative name servers for the domain *foo.bar*, the recursive query could be forwarded by name server A to any one of these authoritative name servers. *How does this affect the accuracy of King?*
4. Our techniques leverage the DNS infrastructure in unanticipated ways. In particular, while gathering a measurement, King relies on having third-party DNS servers generate and answer queries on our behalf, but without their explicit cooperation. *Does King introduce any new security concerns?*

A. How does King find name servers that are close to specific end hosts?

Authoritative name servers for a domain can be found by querying the DNS system for name server (NS) records associated with the domain name. It is a fairly common practice in the Internet to collocate authoritative name servers for a domain close the hosts in that domain. If there are multiple authoritative servers, in practice most of them tend to be placed geographically and topologically close to the hosts within the domain, and King can pick an appropriate server from among them using heuristics we will describe below. However, there do exist exceptions when all the authoritative name servers for an end host are located far away from it (e.g., for ISPs such as AOL). As our detailed studies in Section IV will show, such exceptions are not the common case.

King may be presented with either a DNS host name or an IP address as an end point. Given a DNS name for a host, King picks the authoritative name servers of the host's domain suffix. Given an IP address for a host, King picks the authoritative name servers for its corresponding *in-addr.arpa* domain name. The *in-addr.arpa* domain was specifically created to allow easy reverse lookups (i.e., mappings of IP addresses to host names); to find the name of the host with IP a.b.c.d, one can simply look up the name d.c.b.a.in-addr.arpa.

In Figure 3, we show the authoritative name servers for host with IP address 128.83.139.8. The *in-addr.arpa* name corresponding to this IP address is *8.139.83.128.in-addr.arpa*, and the domain containing this name is *83.128.in-addr.arpa*. The host name corresponding to this IP address is mars-needs-coffee.cs.utexas.edu. As seen in Figure 3, there are four authoritative name servers for the domain, three of which are have names in the utexas.edu domain and one in the uiuc.edu domain.

Type of Resource	Domain Name	IP Address
end host	mars-needs-coffee.cs.utexas.edu	128.83.139.8
authoritative name server	cs.utexas.edu	128.83.139.9
authoritative name server	chisos.ots.utexas.edu	128.83.185.39
authoritative name server	dns.hpc.utexas.edu	129.116.206.1
authoritative name server	dns2.cso.uiuc.edu	128.174.5.104

Fig. 3. The output from a reverse lookup for 128.83.139.8. King selects name server 128.83.139.9 as being the closes to 128.83.139.8, using a combination of DNS suffix matching and IP address prefix matching heuristics.

From these four authoritative name servers, King chooses the closest server to the original host 128.83.139.8 using the following two heuristics. First, King selects name servers whose names share the longest common suffixes with the host name. Second, from the servers selected by the first rule, it chooses the name server whose IP address shares the longest common prefix with the host's IP. The first rule helps to eliminate name servers replicated in other domains, while the second rule gives preference to name servers that reside in the same network as the end host. When applied to our example in Figure 3, the first rule would select the 3 name servers in utexas.edu domain over the name server in uiuc.edu and the second rule would choose the name server with IP address 128.83.139.9 from among the 3 utexas.edu name servers.

B. How often can we find a name server near one of the end hosts that is willing to resolve recursive DNS queries from arbitrary hosts?

Sometimes domain administrators impose access control on name servers, configuring the server so that that the recursive queries from non-local clients are refused. Because King relies on being able to issue recursive queries, we conducted a measurement study that consisted of probing the authoritative name servers of a large number of web servers and home user clients to check if they support recursive queries. We selected 8,306 web servers randomly from a previously studied list of servers [13], and 16,695 Napster clients that were observed in a previous measurement study [18]. Most of the Napster clients are behind modems, cable modems or DSL lines.

We found that 72% of the authoritative name servers associated with web servers support recursive queries from remote clients, while 76% of Napster clients' authoritative name servers do the same. Additionally, 76% of the web servers and 79% of the Napster clients were found to have at least one authoritative name server that supports recursive queries from any arbitrary host. Data from independent studies done at MIT [10] and AT&T [11] (see Section IV-D and Table 6 in [10] and Section 4.1 in [11]) also confirms that a very high percentage of name servers resolve recursive queries from remote clients. Because King only requires one name server out of the two selected to support recursive queries, we expect King to succeed in about $1 - (1 - 0.75)^2 = 0.9375$ or 94% of time. King reports an appropriate error when it fails.

C. When there are multiple authoritative name servers for a domain, a recursive query can be forwarded to any one of these name servers. How does this affect the accuracy of King?

The latency estimated by King is the latency between name server A and the name server to which A forwards the query for resolution. In Section II-A, we described how King can pick appropriate name servers A and B from the sets of multiple authoritative name servers. However, when King issues its recursive query to its selected name server, it is that name server that decides the other name server to which it forwards the query. If there are multiple authoritative name servers besides B, queries might be forwarded by A to any of these different servers, and the latency estimates might not represent latencies between the end points selected by King. This could lead to inaccurate estimates, especially when King issues multiple queries between the same end hosts to improve its accuracy.

Our evaluation in Section IV-A shows that this inaccuracy introduced by the query being forwarded to different name servers is surprisingly small. On further investigation, we attribute this to two common Internet practices. First, while most domains have 3-5 authoritative name servers, at most one of them tends to be placed in an external domain. In many cases, even this external domain is topologically close to original domain. Second, given the IP address of the host, King uses name servers from the *in-addr.arpa* domain; the name servers in this domain tend to be less replicated outside their domain, probably because IP to name bindings are not considered as essential as name to IP bindings. Thus, even when the queries are sent to different name servers, King tends to report identical latencies as all of them are typically located near each other.

However, for applications desiring improved accuracy from King, we have discovered a novel method through which a client can "trick" a name server to forward a recursive query to an authoritative name server of the client's choice.

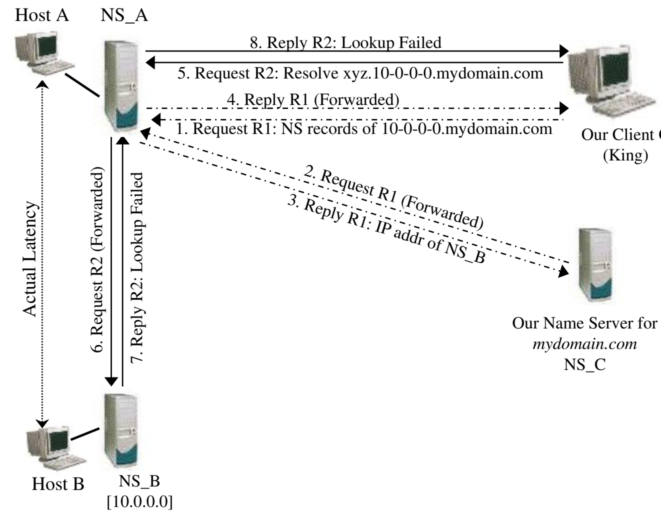


Fig. 4. The sequence of DNS packets used by King to "trick" name server A into directly querying name server B.

The basic idea is to fool name server A into believing that name server B is the authoritative name server for a domain owned by the client C (Figure 4). Suppose, for example, that client C owns *mydomain.com*. Let NS_C denote the authoritative name server for this domain, and assume the IP address of name server B is 10.0.0.0. First, C issues a recursive query R1 to A to resolve name servers for domain *10-0-0-0.mydomain.com*. This subdomain belongs to *mydomain.com*, and the query is forwarded to our name server NS_C. Since we control NS_C, we program it to reply that 10.0.0.0 (i.e., name server B) is the authoritative name server for the sub-domain *10-0-0-0.mydomain.com*. Name server A caches this reply, and forwards it back to client C.

Any subsequent recursive query through name server A for a name belonging to this sub-domain (e.g., foo.10-0-0-0.mydomain.com) will now be forwarded directly to name server B. As B is not actually the authoritative name server for this domain, B replies with an error or a pointer to the root DNS name servers. Name server A now realizes that the lookup failed, and reports the failed lookup to client C. Thus, we achieve our goal of forcing name server A to query name server B directly.

One potential wrinkle with this method is that when B returns an error to name server A, depending on its configuration A may retry the query several times. If so, the latency estimated by King would be a factor K too large, where K is the number of attempts made. One simple way to calculate K is to trick name server A to query NS_C by issuing a bogus name from *mydomain.com*; since we control NS_C, we can observe the number of times a query is retried. Our results indicate that this K typically varies between 1 and 4 for most name servers.

In the common case, however, the authoritative name servers for a domain are all collocated, and this complex "trick" is unnecessary. Hence, for the rest of this paper, we confine our results to those obtained using the simpler method, in which we do not attempt to influence the authoritative name server that is selected by A.

D. Does King introduce any new security concerns?

Although King makes use of the DNS infrastructure in a previously unanticipated way, King does not exploit any vulnerabilities or cause DNS servers to behave in ways that contradict their configuration policy. Furthermore, we don't believe King introduces new vulnerabilities: the tool runs on a client's machine, and its only interaction with external hosts is through DNS queries that the tool itself generates.

There are, however, two potential concerns that should be raised. First, if King becomes popular, it is conceivable that the average load presented to DNS servers could increase. Given that there are several DNS queries generated per web request on average [10], we believe web-induced DNS traffic will dwarf King-induced traffic, even if King becomes popular and widely used.

A second concern is that the complex technique proposed in Section II-C, in which we control which authoritative name server is selected when a recursive query is answered on our behalf, is a form of DNS cache poisoning. While this is true, we only introduce bindings in other DNS servers for names under a domain of our control, and hence our "poisoning" is benign.

In the following section of this paper, we describe related techniques to estimate latencies of Internet paths, and compare them with King.

III. Comparing King to Existing Techniques

IDMaps [6] is the most popular technique to estimate latencies between end hosts in the Internet today. In the IDMaps architecture, end hosts called *tracers* are deployed at strategic locations in the Internet. Each tracer measures the latency between itself and a set of Internet hosts; this set typically includes those hosts in nearby IP address prefixes. Tracers also measure latencies between one another. All measured latencies are sent to a set of dedicated servers, called *HOPS servers*, that use the data to build a virtual topology of the Internet. HOPS servers estimate the latency between two hosts as the sum of the latencies between the hosts and their nearest tracers, plus the distance between the tracers. Any client can query a *HOPS* server to find the latency between two hosts.

GNP [12] is a relatively new technique to estimate the latency between end hosts. A small set of dedicated hosts, called landmarks, first select their own coordinates in a chosen geometric space. These coordinates are then disseminated to any host that wants to compute its own coordinates. Hosts measure the latency between themselves and the landmarks, and from this extrapolate their own coordinates. Given the coordinates of two end hosts, one can then compute the latency between them as a function of the distance between their coordinates. At the time of writing of this paper, work on evaluating the accuracy of GNP at the scale of the wide-area Internet is still in progress, but evaluation over a small number of hosts has demonstrated that GNP is often more accurate than IDMaps.

GNP and IDMaps are similar to each other in that they measure latencies along a subset of Internet paths, and extrapolate these measurements to estimate the latency of an arbitrary Internet path. The primary architectural difference between them is that GNP shifts the complexity of computing distances between landmarks and end hosts from the landmark servers to the end hosts themselves.

Other techniques that could be used to estimate latency include the now defunct *loose source routing* [4], IP to geography mapping techniques [15], [2] and proprietary technologies such as Traceloop [23]. Traceloop uses a large community of *test points* that are provided by volunteer hosts that run the Traceloop software. The latency between any two end hosts is estimated by finding test points close to the end hosts, and measuring the latency between the test points themselves. King cleverly recruits DNS servers as unwitting test points, thus avoiding the need for explicit cooperation from volunteers.

King is significantly different from any of the above techniques, which makes it more suitable for certain applications and less suitable for others. We compare King to GNP and IDMaps with respect to four properties that that are essential for a latency estimation tool to be suitable for use in measurement studies, namely *accuracy*, *speed of estimation*, *latency estimation overhead* and *ease of use*. Our discussion limits its scope to each technique's applicability to measurement studies; we do not reflect the relative merits of the GNP and IDMaps when applied to other applications such as closest server selection, overlay network multicast [3] and content addressable overlay networks [16], which have significantly different requirements for the accuracy and overhead.

Accuracy: Our evaluation will demonstrate King has significantly better accuracy than IDMaps, and although we have not been able to perform a direct comparison with GNP, we expect King estimates to be better than GNP estimates as well due to the differences in the techniques. Both IDMaps and GNP rely on models for the Internet topology to extrapolate latencies, and these models do not account for the complex routing policies and peering agreements between ISPs in the Internet. Previous studies have shown that such models are violated frequently in practice [19], [22]. In many cases the Internet path between the end hosts and the Internet path between the name servers overlap significantly (see Section IV-B.2), which enables King to account both for the complexities of Internet routing paths and transient network properties such as congestion.

Ease of use: King is simpler to use than IDMaps, in that it leverages the existing DNS and does not require the deployment of any additional infrastructure. While it is reasonable to argue that the deployment of the new infrastructure required by IDMaps is only a one time cost, maintaining the infrastructure would require the constant updating of its database of IP prefixes and deployment of additional servers as the Internet or the client population using IDMaps grows. While GNP requires lesser infrastructure, GNP does require the cooperation of end hosts to estimate and share their coordinates relative to an agreed-upon set of landmarks. While hosts participating in a particular system might be willing to share their coordinates amongst themselves, we see no incentive for an end host to share its coordinates in general, as would be required for measurement studies.

Speed of estimation: To estimate latency using IDMaps, a client queries any public HOPS server. To use GNP, the host performing the measurement must discover the coordinates of the end hosts. Assuming that the coordinates of all end hosts could be stored at a collection of servers, GNP will require a single query to these servers. King requires a recursive DNS query to be issued to the name server near an end host, and a second query or ICMP ping to discover the latency to the name server itself. In practice, King takes 3-4 such estimates for higher accuracy. All of the techniques are fast, although King probably takes slightly longer to produce an estimate.

Latency estimation overhead: King generates 2 DNS queries per latency estimate between a pair of end hosts. Assuming King averages estimates to obtain higher accuracy, King will generate 6 DNS queries in total, three to each of name servers near the two end hosts. Each DNS query is a UDP request packet. We feel that this is an acceptably small load on name servers in the Internet today, most of which see light loads. Compared to King, IDMaps and GNP may involve less overhead, since estimates are produced by offline extrapolation from the previously measured latencies of a subset of Internet paths. By relying on extrapolation, IDMaps and GNP trade away accuracy in return for lower estimation overhead.

We believe that King can be used to complement GNP and IDMaps; rather than deploy tracers at various places to measure latencies of Internet paths, IDMaps could use King to estimate the latencies of its paths. Similarly, GNP could use King to compute the coordinates of any host without explicit cooperation.

IV. EVALUATION

In this section, we present a detailed, quantitative analysis of our tool. Our measurements and analysis are focused on understanding two properties of King:

Accuracy. Our first set of experiments are designed to quantify the accuracy of the tool under a variety of scenarios. We present measurements that compare the accuracy of King's latency estimates to "true" latency estimates obtained from public traceroute servers. We also compare the accuracy of King to that of IDMaps. Furthermore, we show the relationship between accuracy and path length, and then demonstrate that King estimates preserve order.

Sources of error. Our second set of experiments are designed to investigate sources of error in King estimates. We demonstrate that application-level latency induced by name servers is negligible, and that King is successfully able to find name servers that are topologically close to end hosts. Finally, we demonstrate that King is able to predict when its own measurements are likely to be inaccurate.

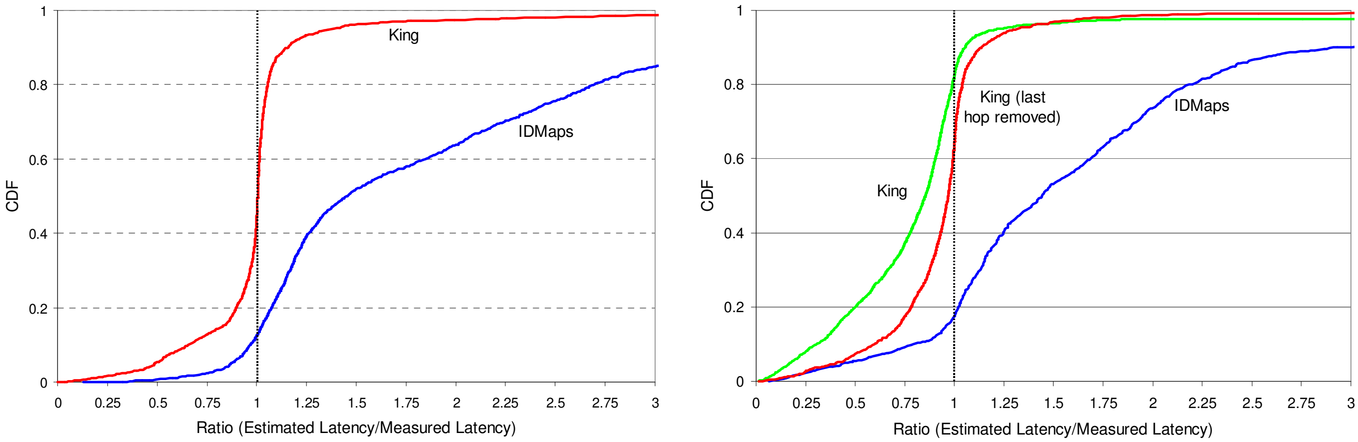
A. Accuracy

To investigate the accuracy of King, we needed to be able to compare King's estimates to directly measured latencies between a large number of hosts around the Internet. To accomplish this, we made use of the large number of public traceroute servers accessible from *www.traceroute.org*; using these, we were able to gather direct latency measurements between the traceroute servers and a large number of Internet end hosts.

We gathered measurements from the traceroute servers to two groups of end hosts: 8,306 web servers selected randomly from a previously studied list of servers [13], and 16,695 Napster clients that were observed in a previous measurement study [18]. While it has been verified that most of the Napster clients are behind modems, cable modems, or DSL lines, we assume that a majority of the web servers have high-bandwidth Internet connections. For the estimates reported here, we configured King to make 4 latency estimates between every pair of end hosts for higher accuracy. To compare the accuracy of King to that of IDMaps, we obtained the latency estimates for the same pairs of end hosts by querying the publicly available IDMaps servers at *www.closestserver.com*.

A.1 Overall Accuracy

We selected 50 traceroute servers and 50 web servers at random, and gathered traceroute measurements (T), King



(a) (b)

Fig. 5. Accuracy of King estimates: (a) traceroute servers to web servers, (b) traceroute servers to Napster clients.

estimates (*K*), and IDMaps estimates (*I*) for the latencies between every pair of hosts. In Figure 5(a), we plot the CDFs of the ratios of King and IDMaps estimates to those of the traceroute estimates (i.e., $\frac{K}{T}$, $\frac{I}{T}$).

If our tool were ideal, the curve would be a vertical line at $X = 1$. In practice, two-thirds of King estimates are within an error of 10% of the actual value, while as high as three-quarters of the estimates are within 20% error margin. In comparison, only 30% of the IDMaps estimates are within a similar error range. We believe that this difference in accuracy is because IDMaps extrapolates latency estimates of a subset of Internet paths, while King directly measures paths. IDMaps tends to overestimate latencies, while King is slightly biased towards underestimation. We explain this phenomenon later in this section.

Next, we selected another set of 50 traceroute servers and 1500 Napster clients. For each Napster client, we selected 3 out of the 50 traceroute servers at random, thus forming a set of $3X1500 = 4500$ pairs. Once again, we gathered traceroute (*T*), King (*K*), and IDMaps (*I*) estimates for these pairs of end hosts and computed the ratios $\frac{K}{T}$, $\frac{I}{T}$. Napster clients tend to have a very large variation in last hop latency, typically ranging from tens to hundreds of milliseconds, as they connect through modems, cable modems, or DSL lines. In contrast, most name servers are well connected and do not suffer from large or variable last hop latencies. Even when the name servers and their end hosts are geographically collocated, the last hop latency of Napster clients can affect the accuracy of King estimates significantly. Accordingly, we also computed the ratio of King estimates (*K*) to traceroute estimates excluding the last hop latency (*T-1*), i.e., $\frac{K}{T-1}$.

In Figure 5(b), we plot the CDFs of the three ratios described above. Excluding the last hop latency, the accuracy of the latency estimates by King for Napster clients is similar to that of the estimates between web servers in Figure 5(a). However, including the last hop latency makes the accuracy noticeably worse. This is a deficiency of King tool; however, the accuracy is still fairly high, and for a number of practical applications (such as closest server replica selection, or many measurement studies), the last hop latency can safely be ignored.

It is also interesting to note that the large last hop latency for Napster clients has the effect of artificially boosting the accuracy of IDMaps estimates. This can be observed by comparing the IDMaps curves in Figures 5(a) and 5(b). Even with this advantage, King still tends to be more accurate than IDMaps.

We investigated a few cases when the latencies in Figure 5(a) were heavily underestimated by King to find the cause of its bias towards underestimation, and we discovered two primary causes. First, not all web servers were well connected as we presumed. In fact, a few of them were behind large latency satellite links. Second, in a few cases, the authoritative name servers were located far away from the end hosts.

A.2 Consistency of Estimates

The consistency of estimates across time is an important characteristic of any measurement tool. King is potentially susceptible to inconsistency when there are multiple authoritative name servers for a domain. To test King's consistency, we compared estimates (K_1, K_2) taken an hour apart from one another. We also obtained traceroute measurements (T_1, T_2) separated by an hour. In Figure 6 we plot CDFs of the ratio of King estimates over time and traceroute measurements over time, i.e., $\frac{K_1}{K_2}$ and $\frac{T_1}{T_2}$. For a tool with perfect consistency, the curve would be a vertical line at $X = 1$. As can be seen from the graph, the consistency of King estimates and traceroute measurements are

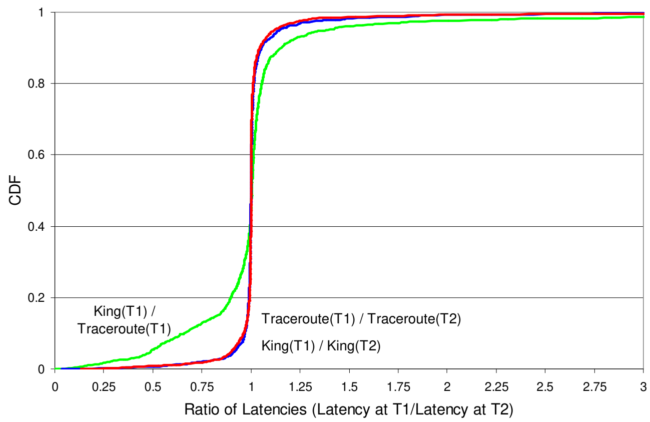


Fig. 6. Consistency of King estimates over time, compared to the consistency of traceroute measurements over time.

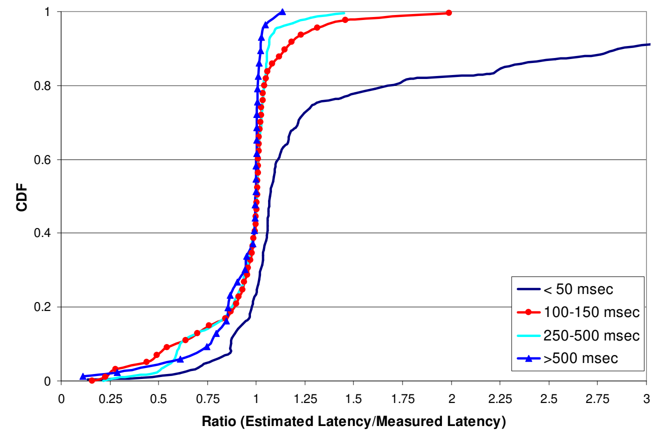


Fig. 7. Accuracy of King estimates as a function of path length.

indistinguishable.

Until now, we have presumed that traceroute measurements represent accurate latency. As Figure 6 shows, even traceroute measurements vary due to transient network conditions. We superimpose the curve showing accuracy of King estimates (i.e., $\frac{K_1}{T_1}$) on the graph to demonstrate systematic differences between King estimates and traceroute. Although our tool clearly underestimates a few latencies, the majority of King estimates match the traceroute measurements.

A.3 Accuracy as a Function of Path Length

A 5 millisecond error in an estimate for a path whose latency is 50 milliseconds accounts for 10% error, while the same 5 millisecond error in an estimate for a path whose latency is 500 milliseconds accounts for 1% error. To examine if the accuracy of King varies with the path latency, we divided the King estimates that we gathered between traceroute servers and web servers into various latency classes (as measured by traceroute). We computed the accuracy of the estimates in each class by calculating the ratio of their King latencies to the actual traceroute measurements; the CDFs of these ratios is shown in Figure 7.

As can be expected, most latencies under 50 milliseconds are overestimated; in such cases, even small errors in estimates translate to large relative errors. As latency increases, high latency last hops lead to increasing underestimation.

A.4 Preserving Order

Preserving the order of latency estimates between pairs with a common end host is important for any latency measurement tool; without preserving order, the tool would not be useful for applications such as closest server selection. To test whether King estimates preserve order, we ranked a set of end hosts based on their actual latencies to a given *reference host*. We also rank these hosts based on their King latencies to the reference host, and then compare the two rankings. The two rankings would be identical for a tool that perfectly preserves order. Even though King has high accuracy in general, a tool could have poor accuracy yet strong order preserving characteristics. For example, our analysis of King has already demonstrated that if a path has a large last hop latency relative to the total path latency, King will produce an underestimate. However, if this large last hop latency is present in all estimated paths, it will not affect the order of the estimates.

Let the two sets $R_1 = \{r_{1,1}, r_{1,2}, r_{1,3}, ...r_{1,n}\}$ and $R_2 = \{r_{2,1}, r_{2,2}, r_{2,3}, ...r_{2,n}\}$ denote the rankings of n hosts based on their actual and estimated latencies from a given reference host, respectively. Note that hosts *closer* to the reference host are assigned *higher* ranks. We used three different metrics to judge the order preserving characteristics of King.

Rank Correlation Coefficient: The rank correlation coefficient is a popular metric that captures the strength of association between two rankings, and is computed using the following formula: $\frac{\Sigma_{i=1}^{i=N}(r_{1,i}-\bar{r_1})(r_{2,i}-\bar{r_2})}{\sqrt{\Sigma_{i=1}^{i=N}(r_{1,i}-\bar{r_1})^2(r_{2,i}-\bar{r_2})^2}}$, where $\bar{r_1}$ and $\bar{r_2}$ denote the averages of the rankings in R_1 and R_2 respectively. This coefficient can vary between -1 and 1; -1 implies a strong negative correlation, while +1 implies a strong positive correlation.

Ranked Accuracy Percent: While the rank correlation coefficient captures the association between the two rankings as a whole, for most applications the correlation between the subsets of highly ranked hosts is more important. Let R_1^k and R_2^k denote the subsets of k highest ranked elements selected from the sets R_1 and R_2 respectively. The rank accuracy percent for k highest ranked hosts captures the number of hosts shared by both subsets, and can be

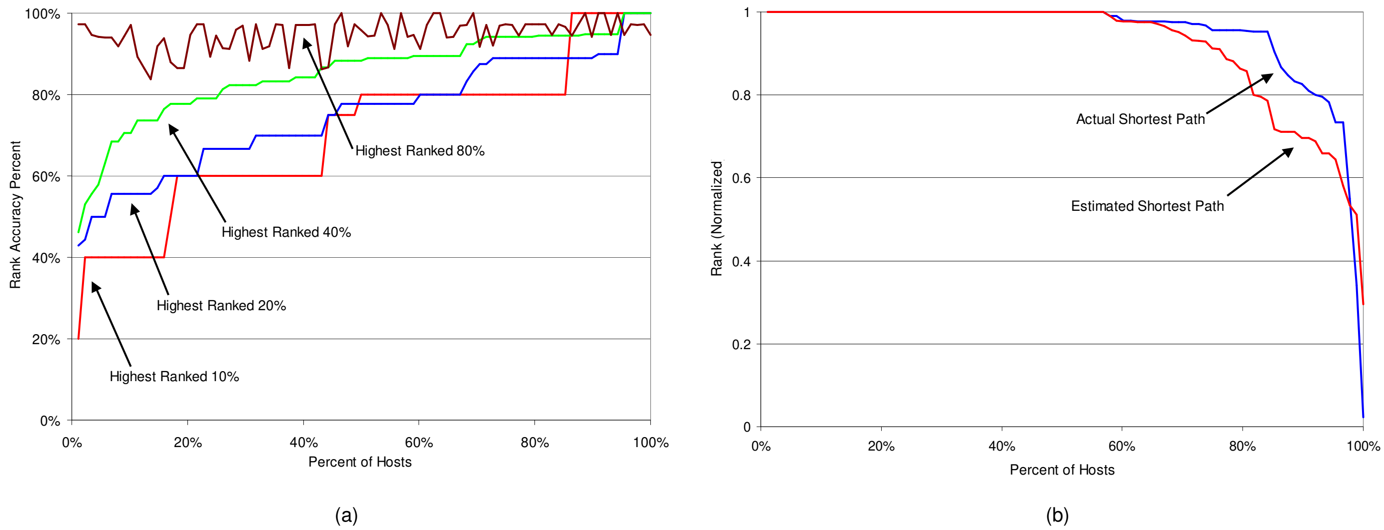


Fig. 9. (a) The rank accuracy percentages for various subset sizes of highly ranked hosts. (b) The normalized true rank of King's best ranked host, illustrating the quality of shortest paths selected by King.

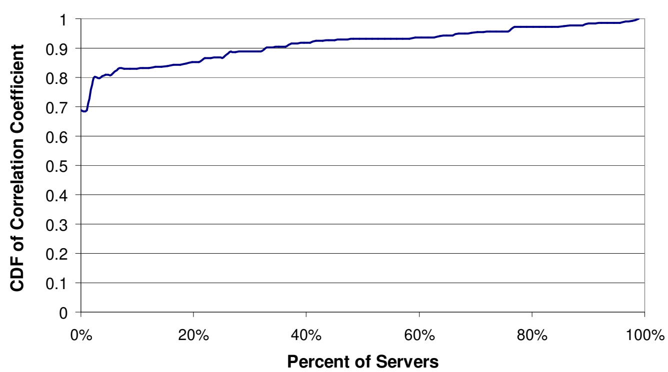


Fig. 8. CDF of the rank correlation coefficient for King estimates between traceroute servers and web servers.

computed by the formula $\frac{n(R_1^k \cap R_2^k)\cdot 100}{k}$, where $n(S)$ denotes the number of elements in set S.

True rank of King's closest-ranked: Content providers might wish to direct their clients to server replicas with best rank as estimated by King. In such a scenario, a metric of interest is the true rank of King's closest-ranked host. Mathematically, this is k, where $r_{1,k} = r_{2,1}$.

For the purpose of this study, we reused our measured and estimated latencies between traceroute servers and web servers. For each of the 50 traceroute servers, we ranked the 50 web servers using traceroute measurements and using King estimates. Similarly, for each of the 50 web servers, we ranked the 50 traceroute servers.

Figure 8 shows the rank correlation coefficients for our measurement sets. Ideally, one would want the correlation coefficient to be +1 for all the hosts. We observe that the correlation coefficient for almost all hosts is greater than $+0.8$, suggesting that King is excellent at preserving order.

In Figure 9(a), we illustrate the rank accuracy percent for varying subset sizes of highly ranked hosts. Specifically, we plot curves for 5%, 10%, 20% and 40% of the total number of hosts. These plots demonstrate that there is significant overlap among the hosts highly ranked by traceroute measurements and King estimates. For applications such as topologically sensitive overlay construction, in which it is necessary to pick hosts from a large overlay that are near to a newly added host, King estimates can be used to prune the set of choices.

Figure 9(b) shows the true rank (normalized to a scale of 1) of King's best ranked host. Given this normalization, perfect estimates from King would lead to a horizontal line at $Y = 1$. In as high as 60% of the cases, the closest host selected by King is same as the closest host selected using traceroutes, while in as high as 80% of the cases the host selected using King lies among the closest 20% of hosts. In this graph, we also show the rank estimated by King for the host with the best true rank. In about 90% of the cases, the estimated rank of the host with the best true rank is among the highest 20%. However, there do exist cases when King's choice proves to be a really bad one. These cases mostly correspond to the scenarios when all name servers picked by King are located far away from their end host. In general case, King would work well for closest server selection applications, but occasional errors could occur.

A.5 Summary

We have presented a detailed evaluation of the accuracy of King; our evaluation has demonstrated that King is consistent, fairly accurate, and it has excellent order preserv-

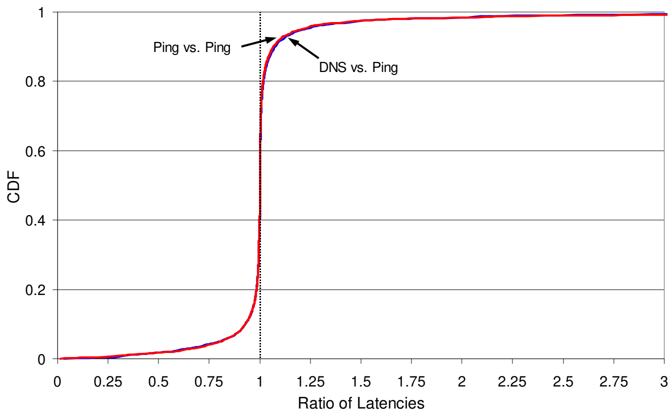


Fig. 10. CDF of the ratio of the query latency to actual latency for name servers (DNS vs. Ping), and CDF of the ratio of actual latencies measured at two different times (Ping vs. Ping). This graph demonstrates that DNS servers do not introduce appreciable application-level latency.

ing characteristics, all of which make it suitable for a large number of applications. One source of error in King estimates arises because King does not account for large last hop delays encountered by many home users; similarly, King estimates are inaccurate in the rare case when all the authoritative name servers are far away from end hosts. Next, we investigate these and other potential sources of error in King.

B. *Sources of Inaccuracy and Error*

There are several potential sources of inaccuracy and errors with King. In this section of the paper, we attempt to quantify them.

B.1 Latency Introduced by Name Servers

King relies on being able to issue DNS queries to name servers in order to deduce latency. One possible source of error is application-level latency introduced by the name servers themselves, while processing our queries. To quantify the magnitude of this latency, we selected 17,782 authoritative name servers of hosts obtained by probing the 32-bit IP address space randomly. We issued an iterative DNS query to each of these name servers to resolve a non-existent name and measured the time T taken. This time, which we refer to as *query latency*, includes the latency between our client and the name server as well as the time taken to resolve the query at the name server. We also measured the *actual latency* to each the name server using ICMP pings, at two different times (*T1', T2'*).

In Figure 10, we plot the ratio of query latency to actual latency ($\frac{T}{T1'}$), as well as the ratio of actual latencies at two different times ($\frac{T1'}{T2'}$). Ideally, both plots should be vertical lines at $X = 1$. Two observations can be made from this data: first, there are variations between DNS query times and ICMP ping times. Second, this variation is indistinguishable from the variation introduced by transient network conditions.

This demonstrates that the query resolution time is negligible, and that query latency can be used as a surrogate for actual latency. We also conclude that latency estimates between name servers by King are highly accurate. One particularly effective way to use King in wide-area measurement projects would therefore be to restrict the set of end hosts being measured to name servers, rather than using arbitrary end hosts.

B.2 Distance between End Hosts and Authoritative Name Servers

The overall accuracy of King depends on how well it picks name servers that are close to end hosts. The high accuracy of King estimates shown in Figure 5 indicates that, in general, King does this well. To confirm this, we evaluated the closeness of authoritative name servers and end hosts using metrics similar to those used in earlier studies from AT&T [11] and IBM [21].

In the AT&T study, the authors used four metrics to evaluate the closeness of end hosts and their local name servers: AS clustering, NAC clustering, traceroute deviation, and RTT correlation. In their paper, the authors conclude that AS clustering is too coarse grained, NAC clustering is too fine grained, and RTT correlation is unreliable due to its dependence on the location from which it is estimated. Thus, our evaluation focuses on traceroute deviation. Traceroute deviation involves obtaining the paths to an end host and its name server from a probe point using *traceroute*. The metric is defined as the maximum number of divergent network hops from a probe location to the client and its name server.

Before proceeding, we note that our study concerns itself with the proximity of *authoritative name servers* to end hosts, while the previous studies dealt with *local name servers*, i.e., name servers that end hosts are configured to use. Intuitively, one might expect local name servers to be closer to end hosts than authoritative name servers. However, end hosts are free to choose multiple local name servers in different autonomous systems (ASs) that are far away from the end host; the AT&T study finds that only 69% of end hosts are configured to use a local name server within the same AS. In contrast, end hosts cannot choose their authoritative name servers, and for reasons of convenience, most domain administrators collocate their authoritative name servers and end hosts. We have observed that over 90% of authoritative name servers lie in the same AS

Path to End Host IP (Latency in msec)	Path to Name Server IP (Latency in msec)
128.95.219.2 (0.258)	128.95.219.2 (0.256)
128.95.219.100 (1.292)	128.95.219.100 (1.281)
140.142.150.24 (0.578)	140.142.155.24 (0.544)
198.107.150.2 (0.897)	198.107.150.2 (0.827)
157.130.181.241 (0.844)	157.130.181.241 (0.952)
146.188.201.30 (1.168)	146.188.201.26 (1.25)
152.63.106.233 (2.121)	152.63.106.225 (2.225)
152.63.107.153 (7.25)	152.63.107.149 (6.731)
152.63.2.37 (34.452)	152.63.32.42 (34.712)
146.188.136.153 (34.944)	152.63.9.69 (34.531)
152.63.91.89 (35.308)	152.63.91.85 (35.424)
157.130.160.142 (36.666)	157.130.160.142 (37.108)
166.70.4.10 (36.316)	166.70.4.10 (36.49)
166.70.4.80 (36.932)	198.60.22.2 (41.507)
166.70.154.233 (69.25)	
166.70.154.236 (69.503)	

Fig. 11. An example of the traceroute paths obtained to an end host and its authoritative name server.

as the end hosts themselves.

To estimate how close authoritative name servers are to their end hosts, we used traceroute to attempt to measure paths from the University of Washington to 6,307 web servers and their authoritative name servers, as well as to 13,157 Napster clients and their authoritative name servers. We successfully obtained 2,561 web-server/name-server pairs, and 1,373 Napster client/name-server pairs. This low success rate is due to ICMP message filtering employed by ISPs.

In Figure 11, we show an example of the paths obtained to the Napster client 166.70.154.236 and its authoritative name server 198.60.22.2. These paths converge and diverge multiple times; the terms *disjoint path to end host* and *disjoint path to name server* refer to the paths from the last point of divergence (157.130.160.142) to the end host and name server, respectively. Traceroute divergence refers to the length (in hops) of these disjoint paths.

We plot the disjoint path lengths to web servers, Napster clients, and their authoritative name servers in Figure 12. We ignore the last hop in the disjoint paths to Napster clients, but not in the paths to web servers or to any name servers; this is necessary so as not to confuse elements of the upcoming analysis.

As can be seen, the traceroute paths to $> 90\%$ of the web servers and their name servers diverge by less than 2 hops. This indicates that web servers are typically *very close* to their name servers. However, the median disjoint path lengths for Napster clients and their name servers are much higher at 4 hops, and about 20% of the disjoint paths are longer than 8 hops. This result is fairly consistent with results in earlier studies; however, it is difficult to infer from the hop count metric whether or not the name servers are close to end hosts in terms of latency. Does a median divergence of 4 hops suggest that the Napster clients are close to name servers, or far away? The IBM study [21] concludes that 5 hops of divergence means that that the name servers are *topologically distant* from the end hosts, whereas the authors of the AT&T study [11] are of the opinion that 4 hops of divergence means that the name servers are *quite close* to the end hosts! Rather than make a subjective judgement call, we investigated more detailed characteristics of these disjoint paths.

To understand how large a 4 hop divergence actually is, in Figure 13(a), we plot the latencies of the disjoint paths to Napster clients and their name servers. As high as 65-70% of the disjoint paths are less than 10 ms, which implies that most of the disjoint paths with hop counts up to 4-5 are short. Furthermore, we inspected a few disjoint paths with high latency and hop counts larger than 8, which led us to a surprising conclusion. For all of the above measurements, we took the last point of convergence between two paths as the last router occurring in both paths with the same IP address. However, in today's Internet, a single router can have multiple IP addresses assigned to it, and paths also flow through physically adjacent routers.

Consider the example paths in Figure 11. The paths diverge after router with IP address 157.130.181.241, but reconverge at the router with IP address 157.130.160.142. It is very likely that the divergent routes taken by these paths between the two routers are physically adjacent, and they have nearly identical latencies from the probe machine. Furthermore, the IP addresses of the routers that these parallel paths flow through match up to 24 bits (three octets): 152.63.91.89 and 152.63.91.85 share a 24 bit address prefix and are both 35 ms away from the probe location.

For many of the disjoint paths with high latency or hop count, we observed that the paths diverged and followed routes that are likely to be adjacent, but never reconverged. To capture this adjacency, we conservatively redefined our last point of convergence between two paths as the last routers in either path that share a network prefix of 24 bits (i.e., a three octet match rather than a four octet match) and whose latencies from the probe location differ by less than 5 milliseconds. Figure 13(b) shows the results; as high as 75-80% of the disjoint paths to Napster clients and end hosts have their latencies differ by less than 10 ms.

We conclude that authoritative name servers are quite close to their end hosts in the general case, which helps to explain the observed high accuracy of King estimates.

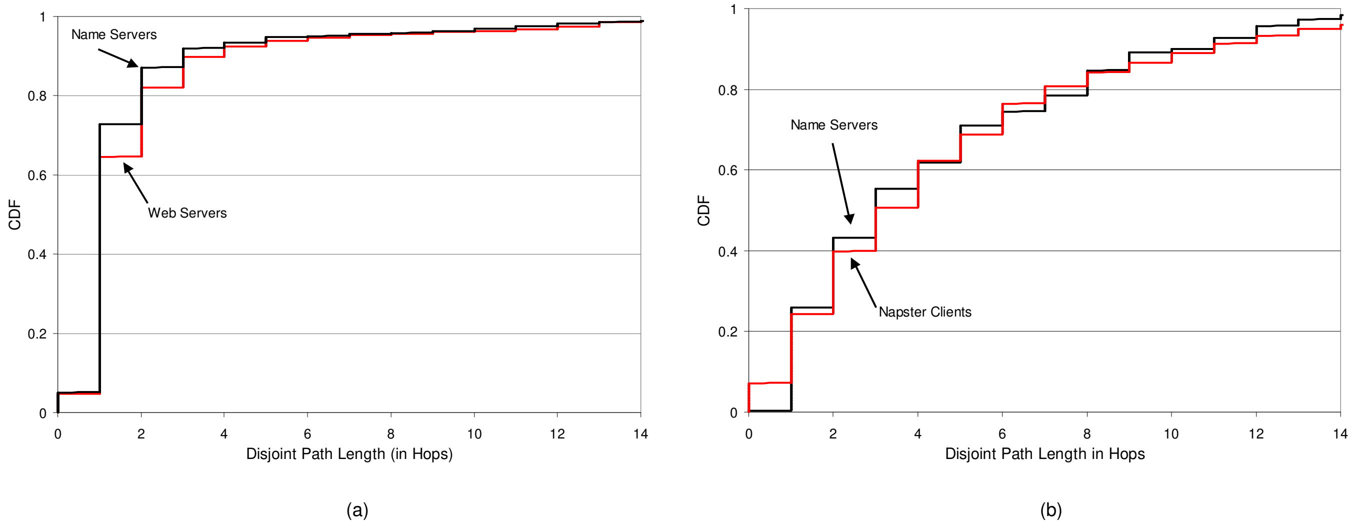


Fig. 12. Hop Counts of divergent paths to hosts and their name servers for proximity evaluation. (a) web servers, (b) Napster clients (excluding the last hop).

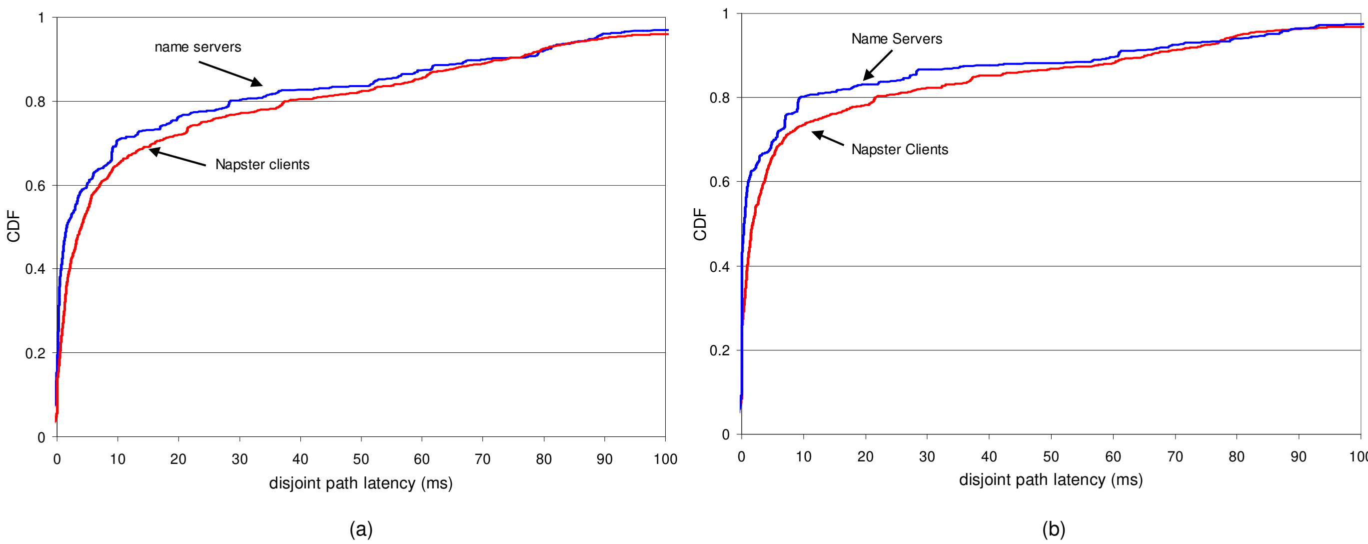


Fig. 13. Divergent paths latency. (a) Napster clients using 4 octet matching, (b) Napster clients using 3 octet matching.

B.3 King's Ability to Predict its own Inaccuracy

Two scenarios account for most of the inaccuracies in King's estimates. First, some end hosts have large last hop latencies which are not accounted for by King. Second, sometimes a few or all of a host's the authoritative name servers may be located far away from the host itself.

We have found that it is possible for King to predict when these sorts of inaccuracies occur. When generating its estimate, King compares the domain names and IP addresses of the authoritative name servers with the domain name and IP address of the end host; the longer the match, the higher the accuracy of the estimate. More specifically, King divides latencies into two categories, based on whether or not the name server it queries shares the same domain suffix as the end host. These two categories are further divided according to whether all, some, or none of the authoritative name servers at the second end host lie within the same domain. Thus, each estimate can be classified into 6 categories of the form X_Y, where $X \in \{full, none\}$ and $Y \in \{full, partial, none\}$. For example, the category *full_partial* implies that the name server queried by King and its end host are in the same domain, but the other end host has some of its name servers in the same domain and some others in different domains.

In Figure 14, we re-plot the King estimates from Figure 5(b), but we classify all estimates as described above, and show three of the six categories. As can be observed, the estimates in category *full_full* do extremely well; in fact, they are indistinguishable from the variation in traceroute estimates that we saw before in Figure 6. In con-

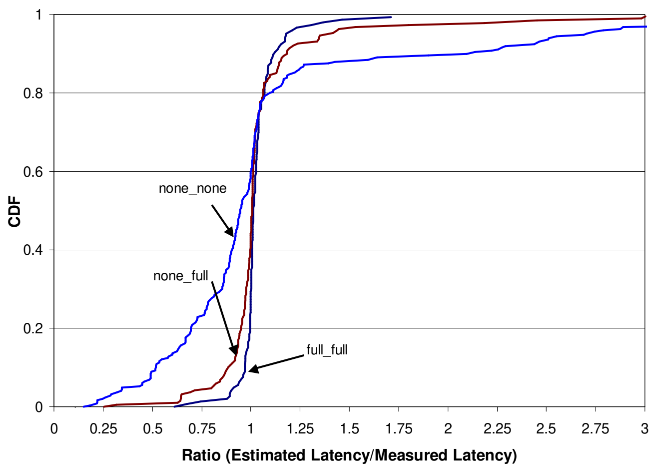


Fig. 14. Accuracy of King as a function of its self-diagnostic ability.

trast, the estimates in the category *none_none* do noticeably worse. This implies that King can use these categories as a self-diagnostic, filtering out estimates that are likely to be inaccurate.

V. Applications of King

King is a general tool that provides the ability to measure latency between arbitrary end hosts. We believe that the simplicity and accuracy of King will motivate a large number of applications, similar to how *ping* and *traceroute* have been used in a large number of unanticipated fashions. However, in this paper, we focus on the wide-area measurement studies that can benefit from King. They fall into three categories.

1. *Those that require a precise distribution of estimates between many end points, but any endpoint may be chosen:* If we have the freedom to select end points, by only considering name servers, King could generate extremely accurate latency estimates. Most of the measurement studies that we have previously mentioned [19], [15], [17] fall under this category.

2. *Those that require a precise distribution of estimates between a large number of specific end points:* For such studies, it is important that the overall distribution of the estimates is accurate even if occasional individual estimates are wrong. For example, King can be used to verify a widely held belief that topology-unaware peer-to-peer overlays (such as that of Gnutella) result in inefficient overlays with links between nodes that are far apart in the Internet.

3. *Those that require precise estimates of specific end points:* This is the most challenging class of measurement studies to satisfy and one has to be cautious when using King for them. Although King's estimates have better precision than existing techniques like IDMaps in general, occasionally estimates have high inaccuracy. King's ability to identify inaccuracies in its estimates could be used effectively in such scenarios.

A. Detour on a Large Scale

As an example of King's usefulness, we duplicated an earlier study (Detour) into the optimality of Internet routing paths [19], [5]. Studies such as this one have been hindered by the fact that they required control over participating end hosts in many different administrative domains, limiting the scale at which the study could be performed. In the original Detour study [5], the authors used only 43 publicly available traceroute servers as their end hosts. Similarly, in the IP2Geo study [15], the authors studied the correlation between the geographical locations of hosts and their IP latencies, but they only controlled a few tens of hosts in cooperating universities.

The Detour study demonstrated that Internet routing paths suffer from inefficiencies, due to non-optimal routing policy decisions at ISPs [22]. Policy decisions, such as private peering agreements, result in alternate paths (referred to as *detours*) that have shorter latency than the directly routed path between two end hosts. We performed the same measurements as in the Detour study, but we used 193x193 pairs of geographically distributed authoritative name servers as our end points, rather than the 43x43 pairs of traceroute servers used in the original study.

We could select our end hosts in two ways. The first option was to select any authoritative name servers that support recursive queries and use the "complex technique" described in Section II-C. The second option was to select authoritative name servers that share at least 3 octets of their IP address with all other authoritative name servers for their domain, and use the simpler version of King. We chose the latter option. For higher accuracy, we collected the latencies between the 193 hosts a total of 8 times on 4 consecutive days, and filtered noisy estimates. Each collection run took under 4 hours using a single client machine.

Figure 15 shows the CDF of the ratio of mean latency of the actual path to the mean latency of the best alternate path. Ideally, one would want the actual path to be the shortest path. However, as we can see, 40% of the paths have more efficient detours. For about 17% of such paths, the alternate paths shaved off 25 milliseconds or more. Our results are similar to those observed in the Detour study [19], but King helped us validate them on a larger scale.

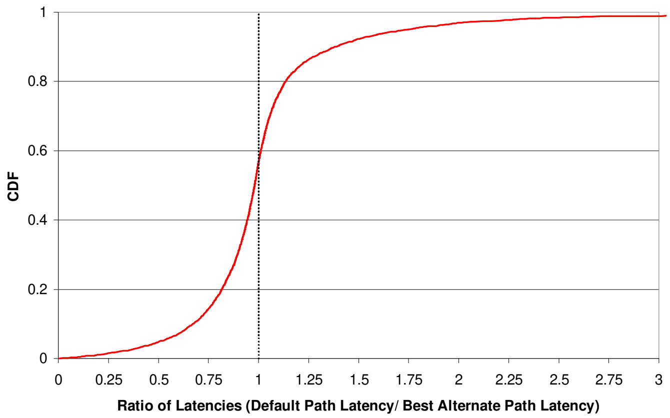


Fig. 15. CDF of the ratio between the mean latency of the default path, and the mean latency for the best alternate path.

VI. Conclusions

In this paper, we presented King, a tool that can accurately and quickly estimate the latency between arbitrary end hosts. Compared to previous approaches, King has several advantages: it does not require the deployment of additional infrastructure, it does not require the active co-operation of measured end points, and it is more accurate as it is based on direct, online measurements rather than offline extrapolation. King works by approximating the latency between two end points by measuring the latency between nearby authoritative DNS name servers using carefully constructed recursive queries. Because King makes use of the existing DNS infrastructure, it scales naturally.

After describing the techniques we used to build King, we presented an extensive evaluation and analysis of its accuracy and consistency. Specifically, our evaluation demonstrated that the accuracy of King is significantly better than the accuracy of IDMaps, and that King tends to preserve order among its latency estimates. Finally, we described a variety of measurement studies and applications that could benefit from our tool, and presented results from one such measurement study.

Acknowledgments

We would like to thank David Wetherall, Mike Swift, Neil Spring, Ratul Mahajan, Sorin Lerner and the anonymous reviewers for their helpful comments on this research and on this paper.

References

[1] N. Brownlee, K. Claffy, and E. Nemeth. DNS measurements at a root server. In *Proceedings of IEEE GLOBECOM '01*, San Antonio, Texas, November 2001.

[2] CAIDA. NetGeo - The Internet geographic database. `http://www.caida.org/tools/utilities/netgeo/`.

[3] Y. Chu, S. Rao, and H. Zhang. A case for end system multicast. In *Proceedings of ACM SIGMETRICS*, June 2000.

[4] D. Estrin, T. Li, Y. Rekhter, K. Varadhan, and D. Zappala. RFC 1940 - Source demand routing: packet format and forwarding specification, May 1996.

[5] S. Savage et al. Detour: a case for informed Internet routing and transport. In *IEEE Micro, pp. 50-59, v 19, no 1*, January 1999.

[6] P. Francis, S. Jamin, C. Jin, Y. Jin, D. Raz, Y.Shavitt, and L. Zhang. IDMAPS: a global Internet host distance estimation service. In *IEEE/ACM Trans. on Networking*, October 2001.

[7] J.D. Guyton and M.F. Schwarz. Locating nearby copies of replicated Internet servers. In *Proceedings of ACM SIGCOMM'95*, August 1995.

[8] T. Hansen, J.Otero, T. McGregor, and H. Braun. Active measurement data analysis techniques. `http://amp.nlanr.net`.

[9] K. Johnson, J. Carr, M. Day, and F. Kaashoek. The measured performance of content distribution networks. In *Proceedings of the Fifth WCW, 2000*, Lisbon, Portugal, May 2000.

[10] Jaeyeon Jung, Emil Sit, Hari Balakrishnan, and Robert Morris. DNS performance and the effectiveness of caching. In *Proceedings of the First ACM SIGCOMM Internet Measurement Workshop (IMW 2001)*, San Francisco, CA, November 2001.

[11] Z. Morley Mao, C. Cranor, F. Douglis, M. Rabinovich, O. Spatscheck, and J. Wang. A precise and efficient evaluation of the proximity between web clients and their local DNS name servers. In *Proceedings of the Usenix Annual Technical Conference, 2002*, Monterey, CA, June 2002.

[12] Eugene Ng and Hui Zhang. Predicting Internet network distance with coordinates-based approaches. In *Proceedings of IEEE INFOCOM 2002*, NewYork, NY, June 2002.

[13] J. Padhye and S. Floyd. The TBIT web page. `http://www.icir.org/tbit/daxlist.txt`.

[14] Jitendra Padhye and Sally Floyd. On Inferring TCP Behaviour. In *Proceedings of ACM SIGCOMM'01*, San Diego, CA, August 2001.

[15] V.N. Padmanabhan and L. Subramanian. An investigation of geographic mapping techniques for Internet hosts. In *Proceedings of ACM SIGCOMM'01*, San Diego, CA, August 2001.

[16] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. A scalable content-addressable network. In *Proceedings of ACM SIGCOMM'01*, San Diego, CA, August 2001.

[17] Sylvia Ratnasamy, Mark Handley, Richard Karp, and Scott Shenker. Topologically-aware overlay construction and server selection. In *Proceedings of IEEE INFOCOM 2002*, NewYork, NY, June 2002.

[18] Stefan Saroiu, P. Krishna Gummadi, and Steven D. Gribble. A measurement study of peer-to-peer file sharing systems. In *Proceedings of Multimedia Computing and Networking 2002 (MMCN '02)*, San Jose, CA, January 2002.

[19] S. Savage, A. Collins, E. Hoffman, J. Snell, and T. Anderson. The End-to-end effects of Internet path selection. In *Proceedings of the ACM SIGCOMM'99*, Cambridge, MA, September 1999.

[20] Stefan Savage. Sting: a tcp-based network measurement tool. In *Proceedings of the 1999 USENIX Symposium on Internet Technologies and Systems (USITS '99)*, October 1999.

[21] A. Shaikh, R. Tewari, and M. Agarwal. On the effectiveness of domain name system. In *Proceedings of IEEE INFOCOM, 2001*, Anchorage, Alaska, April 2001.

[22] H. Tangmunarunkit, R. Govindan, S. Shenker, and D. Estrin. The impact of routing policy on Internet paths. In *Proceedings of IEEE INFOCOM, 2001*, Anchorage, Alaska, April 2001.

[23] Traceloop. The Traceloop home page, November 2000. `http://www.traceloop.com`.