In Proc. of the Third IEEE Workshop on the Architecture and Implementation of High Performance Communication Subsystems (HPCS'95)

Dynamic Server Selection in the Internet

Mark E. Crovella and Robert L. Carter
Computer Science Department, Boston University
111 Cummington St., Boston MA 02215
{crovella,carter}@cs.bu.edu

June 30, 1995

1 Introduction

As distributed information services like the World Wide Web become increasingly popular on the Internet, problems of scale are clearly evident. Three scaling impediments to such distributed information services are excessive server load due to popular documents, wasted network bandwidth due to redundant document transfer, and excessive latency in delivering documents to the client due to the potential for transfers over slow paths. A promising technique that addresses all of these problems is service (or document) replication. However, when a service is replicated, clients then need the additional ability to find a "good" provider of that service.

In many cases, clients may know in advance which service providers are best for them. Such a static server selection scheme is used, *e.g.,* in the distribution of network news using NNTP. However, static server selection is inappropriate in a number of cases. For example: 1) The authors in [3] employ a dynamic protocol to find the best server for a document: the client probes a set of servers who may have replicated the document, as well as the document's home server, and chooses the first to reply as the source of the document. 2) A mobile client using a replicated service will want to find the "best" server to serve the client's requests; the choice will depend on the changing location of the client. 3) Finally, our current interest is in replicating WWW documents. A dynamic server selection mechanism will enable a very flexible, responsive policy for document replication.

In this paper we report on techniques for finding good service providers without *a priori* knowledge of server location or network topology. We only assume that the client has been provided with a list of addresses of servers that provide the required service. We focus on the problem of dynamically selecting a server for replicated documents, but most of our results are applicable to replicated services in general. We consider the use of two principal metrics for measuring distance in the Internet: hops, and round-trip latency. We show that these two metrics yield very different results in practice. Surprisingly, we show data indicating that the number of hops between two hosts in the Internet is *not* strongly correlated to round-trip latency. Thus, the distance in hops between two hosts is not necessarily a good predictor of the expected latency of a document transfer.

Previous work on server selection has emphasized static selection, relying on a previous server discovery step [5], or random selection [2]. Static selection work has used hops as the distance metric, since this metric varies least over long periods. We show in this work that the extra cost at runtime incurred by dynamic latency measurement, as compared to prior static knowledge of hop distances, is well justified based on the resulting improved performance. In addition we show, not surprisingly, that selection based on dynamic latency measurement is preferable to random selection.

The difference between the distribution of hops and latencies is fundamental enough to suggest differences in algorithms for server replication. We show that conclusions drawn about service replication based on the distribution of hops need to be revised when the distribution of latencies is considered instead.

2 Hops Vs. Latency

To characterize the difference between the use of hops and the use of round-trip time as distance metrics, we show empirically measured distributions of the two metrics. We measured the values of these two metrics between a fixed host ("client") on our local network and 5262 hosts ("servers") randomly selected from a list of WWW servers [7].

Figure 1 shows the measured distribution of hops to the set of servers on the left, and shows a quantile-quantile plot of the distribution versus a normal distribution on the right. The figure shows that the data is very nearly a normal distribution, excepting a bump in the 8 hops range and a slightly heavy upper tail. The parameters of the normal distribution are $\bar{x} = 16.6$ and $\hat{\sigma} = 4.0$. Inspection of the raw data shows that the subset

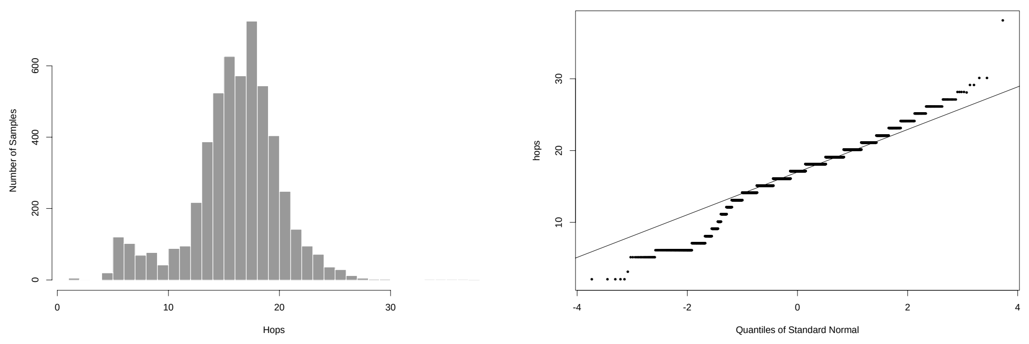

Figure 1: Empirical Distribution of Hops to 5262 Random Servers

of hosts clustered around 8 hops are those accessible via our regional network (NEARnet); the large bell-curve represents hosts reached via a network backbone.

Although we surveyed distances using only one client host, this distribution is remarkably similar to measurements reported in [5] that average over many client hosts ($\bar{x} = 17$ and $\hat{\sigma} = 4.3$). Thus we feel that our use of a single client host in these experiments has not affected our data drastically.

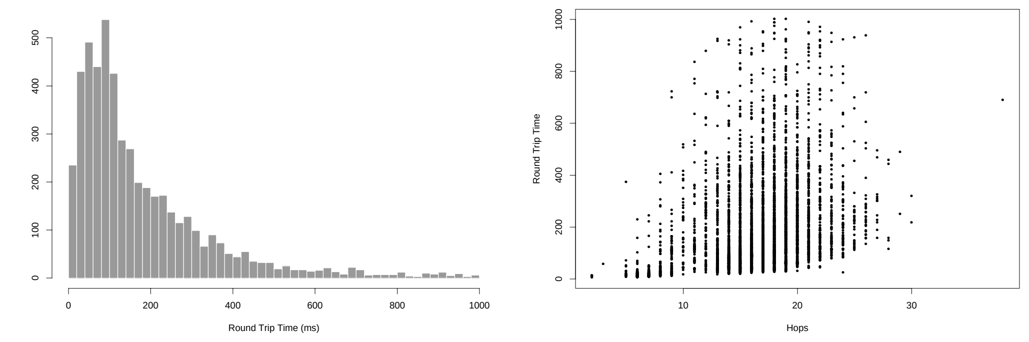

Figure 2: Left: Empirical Distribution of Round-Trip Time to 5262 Random Servers; Right: Scatter plot of Round Trip Time Vs. Hops

The distribution of round-trip latencies (in ms, measured using `ping`) is markedly different, as shown in Figure 2 (left). Whereas the distribution of hops is fairly symmetric about the mean, the distribution of latencies has a median (125) much less than the mean (241), which is more characteristic of probability mass functions like the exponential or Gamma distribution in which a majority of the probability mass is contained below the mean. In addition, the standard deviation of round trip time is nearly twice the mean ($\bar{x} = 241$ and $\hat{\sigma} = 435$); for hops the standard deviation was about a quarter of the mean. These differences have implications for replication policies, as shown in the next section.

Given the differences between these two metrics, it is not surprising that they are not strongly correlated. A scatter plot of the same data is shown in Figure 2 (right). The correlation between hops and round trip time is too weak to be of predictive value; a least squares linear fit to the data has $R^2 = 0.10$, indicating that only 10% of the total variation can be explained by a linear relationship between hops and round trip time. Other authors [6] have found a higher correlation between hops and latency, but under restricted assumptions and still not yielding a strong correlation.

3 Placement of Replicas

The differences between the distributions shown in Figures 1 and 2 have implications for a replica placement policy. A number of replica placement policies have been suggested that attempt to place replicated servers or documents "near" to the clients that will use them (*e.g.,* [6]) This is a natural conclusion if the distance metric used is hops. Because the probability mass of the hops distribution is clustered about the mean, a careful replica placement policy is required to minimize the number of hops between the client and the server. Put another way, a random distribution of replicas throughout the Internet does not decrease the average number of hops between a client and server by much.

However, the same is not true when distance is measured in round-trip latency. Since the probability mass of the latency distribution lies largely below the mean, the effect of randomly distributing replicas throughout the Internet is to sharply decrease the latency between a client and server.

Thus, in terms of round trip latencies, the performance of a replicated server system is likely to be less sensitive to server placement than it would be in terms of hops. To assess the magnitude of this difference, we consider the following service replication strategy. Assume that each of m documents or services is replicated in a different set of n servers; each server set of n is chosen randomly from all of the servers in the Internet. Furthermore, for each service, there is a home server that has a record of the locations of the replicas. When a client requests a service, it asks the home server for the replica list. Using a dynamic server selection strategy, the client then discovers the nearest replica. What will be the average distance between the client and the replica, both in terms of hops and in terms of round trip latency, over all m services?

To answer this question for $m = 1000$ and varying n, we performed the following Monte Carlo simulation using our dataset. At a given replication level n we select n random samples from our empirical hops and round-trip latency distributions. The minimum over n is the result for one trial. The average of m such trials is the final result.

We show the results in Figure 3. On the left we vary the number of replicas from 1 to 20; on the right it is varied from 1 to 200. The y axis in these plots is the distance between the client and dynamically-discovered server as a fraction of the mean value. The figure shows that the distance in hops between the client and server does not decrease rapidly with increasing replication, and never drops much below 25% of the mean. However, the distance between the client and dynamically-discovered server drops very sharply in terms of round-trip latency. The round trip latency l as a function of the replication level n is a very close fit to $l = 0.55/n^{.62}$ ($R^2 = 0.98$) indicating that random server replication is very efficient at reducing round trip latency.

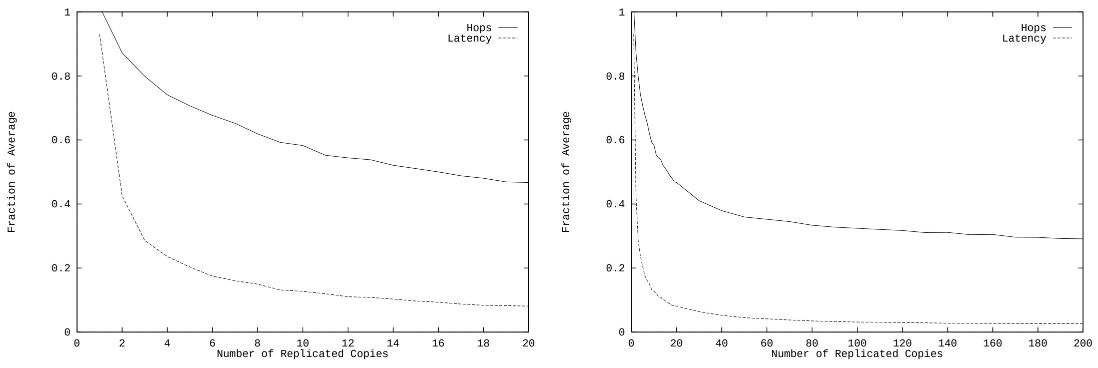

Figure 3: Performance of Random Replica Placement with Dynamic Server Selection

4 Dynamic Server Selection

4.1 The Need for Dynamic Server Selection

The failure of hops as a predictor variable for round trip latency suggests that either variation in link speed or delays due to congestion dominate the round trip time of packets.

To assess the impact of congestion on the distribution of round trip times we collected time series data measuring round trip times to a variety of individual hosts over extended periods. A typical example is shown in Figure 4. On the left is a time series plot of round trip time to a single host measured approximately every 30 seconds over a period of nearly two days; on the right is a histogram of the same data. The host is 19 hops distant and has a mean round trip time of 289 ms, slightly more than the mean values for all hosts presented in the previous section, so it is fairly representative. The histogram shows that there is significant variation in the round trip time to this host, and the time series plot shows that this variation occurs over short timeframes.

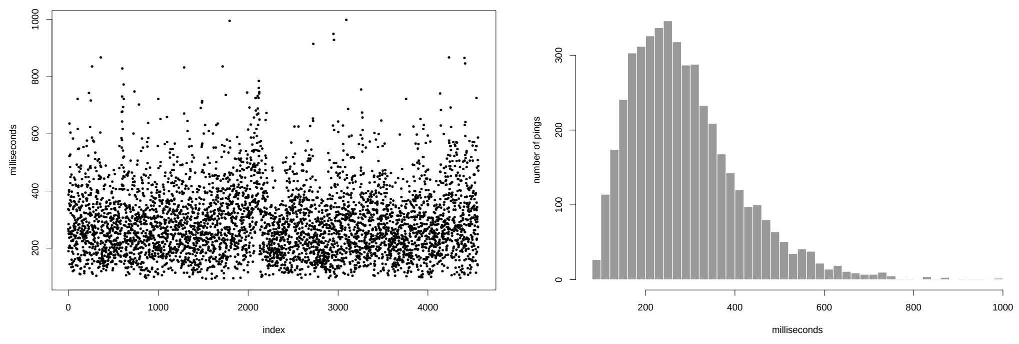

Figure 4: Round Trip Times to a Single Host

As a result of these observations we expect that congestion plays a significant role in the overall distribution of round trip times. This indicates that static policies for server selection are likely to be less effective in general than dynamic policies. We show in the next section initial results confirming this hypothesis.

4.2 Evaluating Dynamic Server Selection

A dynamic server selection strategy such as we envision requires a list of hosts which can provide the given service, and a method for quickly estimating the cost of requesting the service from each host. In the case of WWW documents, we assume that a replication mechanism exists that places copies of WWW documents on proxy servers. Several groups are studying such mechanisms [1, 4]. In addition we assume that it is possible to obtain a list of hosts containing replicas of a particular document (perhaps by querying the home site for the document).

We simulated a dynamic server selection policy for transferring WWW documents using real hosts on the Internet. We identified 10 hosts, each of whose average round-trip latency was approximately equal to the mean we measured for all hosts. Over a 72 hour period we periodically measured the round-trip latency to each host five times, using `ping`. We then measured the transfer time from each host for a document of size 1K, 5K, 10K, and 20K bytes. Using this data we simulated a number of server selection policies: 1) Static, based on geographic distance (Geographical); 2) Static, based on the number of hops (Hops); 3) Dynamic, based on a random server selection (Random); and 4) Dynamic based on the mean of 1,2,3,4 or 5 round trip measurements (Dynamic 1-5). The results are shown in Figure 5.

The figure shows that dynamic policies consistently outperform static policies, and that the difference between static policies and dynamic policies increases with increasing document size. In fact, even random server selection is preferable to choosing any static server for documents larger than 5K bytes.

Comparing methods for estimating transfer cost, the figure shows that all of the dynamic policies yield good results for documents of size 10K bytes or less. However, when estimating the cost of a 20K byte transfer, averaging over four measurements yields better results than one, two or three, while averaging five measurements

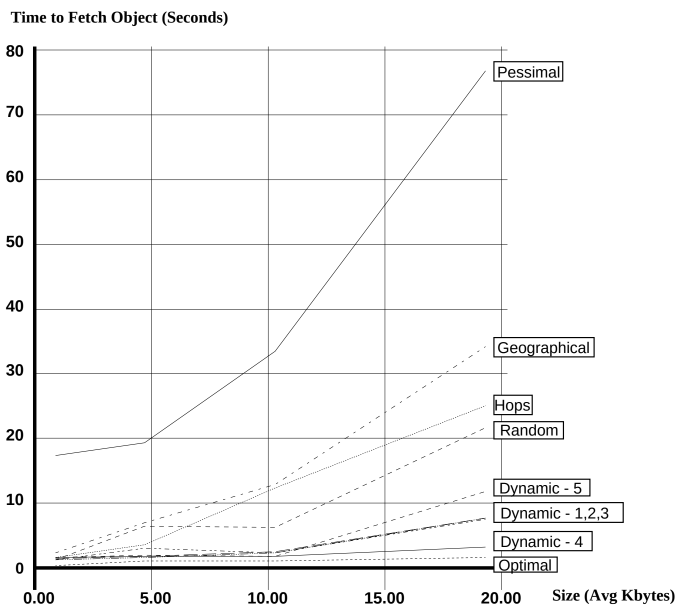

Figure 5: Fetch Times of Static and Dynamic Policies Plotted Against Document Sizes.

yields worse results. This suggests that bandwidth limitations and congestion effects begin to contribute to transfer time as transfer sizes exceed 10K bytes.

5 Conclusion

This work has a number of limitations which are topics of current study. Proper dynamic server selection will require a characterization of the load on a server as well as a measurement of the available bandwidth to each candidate server. We will rely on known techniques for dynamically measuring path bandwidth. In addition, careful study of the time constants and autocorrelation of server path performance is needed, to understand how far into the future we can predict based on present measurement of path characteristics.

Still, these results are suggestive that in an environment of replicated services or documents, dynamic selection of servers holds promise as a effective mechanism to reduce service latency at the client, balance traffic to avoid congestion, and simplify service replication policies.

References

[1] Azer Bestavros. Demand-based resource allocation to reduce traffic and balance load in distributed information systems. In *Proceedings of SPDP'95: The* 7^{th} *IEEE Symposium on Parallel and Distributed Processing*, San Anotonio, Texas, October 1995.

[2] C. Mic Bowman, Peter B. Danzig, Darren R. Hardy, Udi Manber, and Michael F. Schwartz. Harvest: A scalable, customizable discvoery and access system. Technical Report CU-CS-732-94, Department of Computer Science, University of Colorado, Boulder, Colorado, August 1994.

[3] Anawat Chankhunthod, Peter B. Danzig, Chuck Neerdales, Michael F. Schwartz, and Kurt J. Worrell. A hierarchical internet object cache. Technical Report CU-CS-766-95, University of Colorado - Boulder, Mar 1995.

[4] Yasuhiro Endo, James Gwertzman, Margo Seltzer, Christopher Small, Keith A. Smith, and Diane Tang. VINO: The 1994 fall harvest. Technical Report TR-34-94, Harvard University Department of Computer Science, 1994.

[5] James D. Guyton and Michael F. Schwartz. Locating nearby copies of replicated internet servers. In *Proceedings of SIGCOMM '95*, August 1995.

[6] James Gwertzman and Margo Seltzer. The case for geographical push-caching. In *HotOS '94*, 1994.

[7] net.Genesis Corporation. Comprehensive list of sites. Available at `http://www.netgen.com/cgi/comprehensive.`, April 1995.