On the Performance of Quorum Replication on the Internet

Omar Mohammed Bakr
Idit Keidar

Electrical Engineering and Computer Sciences
University of California at Berkeley

Technical Report No. UCB/EECS-2008-141
http://www.eecs.berkeley.edu/Pubs/TechRpts/2008/EECS-2008-141.html

October 31, 2008

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
31 OCT 2008		**00-00-2008 to 00-00-2008**

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
On the Performance of Quorum Replication on the Internet	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **University of California at Berkeley,Electrical Engineering and Computer Sciences,Berkeley,CA,94720-1700**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT
see report

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			
unclassified	**unclassified**	**unclassified**	**Same as Report (SAR)**	**13**	

Acknowledgement

The majority of the hosts we used belong to the RON Project, which is funded by DARPA. We are grateful to Dave Andersen for his dedicated assistance with these machines. We are also indebted to Sebastien Baehni, Danny Bickson, Danny Dolev, Alan Fekete, Rachid Guerraoui, Yuh-Jzer Joung, Mark Moir, Sergio Rajsbaum, Greg Ryan, and Ilya Shanayderman for hosting our experiments on their machines and for providing technical support.

On the Performance of Quorum Replication on the Internet

Omar Bakr
UC Berkeley
ombakr@cs.berkeley.edu

Idit Keidar
The Technion
idish@ee.technion.ac.il

***Abstract*— Replicated systems often use *quorums* in order to increase their performance and availability. In such systems, a client typically accesses a quorum of the servers in order to perform an update. In this paper, we study the running time of quorum-based distributed systems over the Internet. We experiment with more than thirty servers at geographically dispersed locations; we evaluate two different approaches for defining quorums. We study how the number of servers probed by a client impacts performance and availability. We also examine the extent to which cross-correlated message loss affects the ability to predict running times accurately from end-to-end traces.**

I. Introduction

Replication is a fundamental tool in achieving reliability and high availability. There is a cost to keeping replicas consistent: operations need to be disseminated to multiple replicas. However, operations need not be disseminated to all the replicas in order to ensure consistency; it suffices to have operations access a *majority* of the replicas [1], or a collection of replicas that have a majority of *votes* [2]. More generally, replica management can be based on the notion of a *quorum system*. Given a collection of hosts (replicas, servers), a *quorum system* is a collection of sets of hosts, called quorums, such that every two quorums in the collection intersect. Given a known quorum system, it suffices to have each operation access a quorum of the replicas in order to ensure consistency. This approach is employed by numerous systems, e.g., [3], [4], [5], [6], [7], [8], [9].

The simplest and most common quorum system is *majority*, where the quorums are the sets that include a majority of the hosts. A *crumbling wall* quorum system [10] employs much smaller quorums; it has efficient constructions with quorums ranging in size from $O(\log n)$ to $O(\frac{n}{\log n})$. A crumbling wall is constructed by arranging hosts in rows of varying widths, as shown in Fig. 1. A quorum is a union of a full row and one element from each row below the full row. A common metric for quorum systems is their availability, measured as the probability of at least one quorum surviving. The majority quorum system has been shown to be the most available one assuming independent identically distributed (IID) host failure probabilities, and no partitions [11][1], and crumbling walls have been shown to have the highest availability among systems with such a small quorum size [10]. Since on the Internet partitions do occur and crashes are not IID, these results do not always hold [12]. In this paper, we observe that crumbling walls are usually as available as majority, and sometimes even more available.

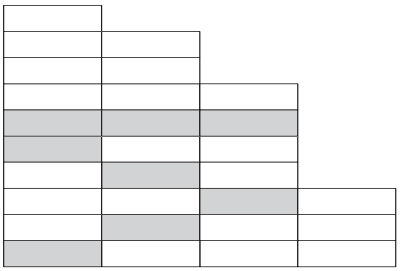

Fig. 1. Crumbling wall with 10 rows of sizes 1,2,2,3,3,3,3,4,4 and 4; one quorum shaded.

We study a simple primitive modeling the communication pattern occurring in quorum-based replicated systems. This primitive has one host, called the *initiator*, gather a small amount of information from a quorum of the hosts. There are many quorums in a quorum system, but each instance of the primitive has to gather information from just one of them. This raises the question of how many hosts to probe, i.e., to send requests to. We call the set of hosts probed by the initiator the *probe set*. One option is to use a *complete probe set* consisting of all the hosts in the system, and wait for any of the quorums to respond. At the other extreme, it is possible to use a *minimal probe set*, consisting of exactly one quorum. In general, it is possible to probe any number of quorums, and wait for one of them to respond. There is a tradeoff in choosing the probe set: smaller probe sets reduce the overall system load, whereas larger probe sets increase availability, and can potentially improve performance. We study the impact of the choice of probe set on running time (response time) and availability; other factors, such as load and throughput, are left for future work.

Our study shows that increasing the probe set beyond a minimal one yields significant performance gains. However, these performance gains taper off beyond a certain probe set size (depending on various factors), after which it is no longer cost-effective to increase the probe set. We further observe that with the majority quorum system, minimal probe sets yield low availability, whereas with crumbling walls they achieve high

[1]This holds for host failure probabilities of up to 0.5.

availability.

We conduct our study by running experiments over the Internet. Our experiments span over 30 widely distributed hosts across Europe, North America, Asia, and the Pacific; at both academic institutions and commercial ISP networks. We present data that was gathered over several weeks. By running our experiments on the actual wide area network, we ensure that our results reflect real factors such as correlated loss. Indeed, we observe a high cross-correlation of message loss. That is, the loss probabilities of messages sent by an initiator to different hosts in a given instance are correlated. In order to illustrate the effect correlated loss has, we devise a simple *estimator* that predicts performance based on the underlying end-to-end network characteristics assuming independent latency variations[2]. We compare our measured results with those predicted by the estimator, and show that analysis relying on end-to-end traces is not a substitute for running experiments in a real network.

In most of our experiments, the hosts communicate using TCP/IP. It was feasible for us to deploy a TCP-based system because TCP is a "friendly" protocol that does not generate excessive traffic at times of congestion. We also include some results from experiments using UDP/IP. For UDP-based experiments, we have implemented a conservative loss detection and recovery mechanism, similar to that of TCP, in order to preserve the "friendliness" property. The TCP and UDP results are virtually identical, except when loss rates are significant, in which case UDP out-performs TCP. Additionally, the UDP measurements allow us to study the extent of cross-correlated loss.

The rest of this paper is organized as follows: Section II discusses related work. Section III describes the experiment setup and methodology. Section IV presents TCP-based tests studying the effect of the probe size set on running time and availability. Section V presents results using UDP, and examines correlated message loss. Section VI concludes.

II. Related Work

A fair amount of work has been dedicated to measuring the routing dynamics and end-to-end characteristics of Internet links[3] [13], [14], [15], [16], [17]. However, such research focuses primarily on point-to-point communication. As we show in this paper, one cannot rely on independent point-to-point traces in order to accurately predict the performance of a distributed system, since such traces do not capture cross-correlation of loss probabilities on different links.

Another recent line of research studies the availability of various services running over the Internet, e.g., content distribution servers [18], peer-to-peer systems [19], [20], and point-to-point routing [16], [17]. As these studies do not consider quorum replication, they are orthogonal to our study.

Although quorum replication is widely used, we are not aware of any previous study of the running time of such systems over the Internet. Moreover, we are not familiar with any study dealing with the impact of probe set sizes on quorum-based systems. The primary foci for previous evaluations of quorum systems were availability and load. Load is typically evaluated assuming minimal probe sets, each consisting of a single quorum. Availability has been studied using probabilistic modeling [11], [21], and Amir and Wool have studied it empirically, over the Internet, in a limited setting consisting of 14 hosts located at two Israeli universities [12]. Note that such studies implicitly assume complete probe sets, as they merely examine whether a live quorum exists. In contrast, our availability study also examines the impact of the probe set size on the probability of a live quorum responding. Peleg and Wool define a related (but different) metric, called *probe complexity*, which is the worst case number of probes required to find a live quorum or to show that none exists [22].

In earlier work [23], we have evaluated the running time of a primitive that gathers information from *every* live host in a system. We found that the performance of such a primitive is highly sensitive to message loss; the presence of a single link with a high loss rate can drastically degrade performance. Follow-up work by Anker et al. [24] studies factors influencing total order protocols in wide area networks, also observing that loss rate and latency variations have a significant impact on performance. In this paper, however, we observe that when only responses from a quorum are awaited, unreliable links can generally be masked, and the impact of message loss is therefore much smaller. Gathering responses from a quorum can therefore be an order of magnitude faster than probing all hosts [25].

Jimenez-Peris et al. [26] argue that when the vast majority of operations are reads, writing to all available hosts is preferable to quorum replication, since it allows reads to be performed locally. However, their study does not take into account loss rates or the high variability of latency in the Internet. Moreover, their preferred strategy, namely writing to all available hosts, assumes that there are neither network partitions nor false suspicions of live servers. These assumptions do not hold in today's Internet.

III. Methodology

A. The Hosts

Our experiments span 36 hosts, as detailed in Table I. Most of the hosts are part of the RON testbed [16]; 24 hosts are located in North America, 6 in Europe, and the rest are scattered in Israel, East Asia, Australia, and New Zealand. No two hosts reside on the same LAN.

B. Server Implementation and Setup

Each host runs a server, implemented in Java. Each server has knowledge of the IP addresses of all the hosts in the system. A `crontab` monitors the server status and restarts it if it is down. We constantly run ping and traceroute from each host to each of the other hosts in order to track the underlying routing dynamics, latencies, and loss rates.

[2]Latency variations most often occur due to message loss.

[3]We refer to the end-to-end communication path between two hosts on the Internet as a *link*.

Name	Description
AUS	University of Sydney, Australia
CA1	ISP in Foster City, CA
CA2	Intel Labs in Berkeley, CA
CA3	ISP in Palo Alto, CA
CA4	ISP in Sunnyvale, CA
CA5	ISP in Anaheim, CA
CA6	ISP in San Luis Obispo, CA
CHI	ISP in Chicago, IL
CMU	Carnegie Mellon, Pittsburgh, PA
CND	ISP in Nepean, ON, Canada
CU	Cornell University, Ithaca, NY
Emulab	University of Utah, UT
GR	National Technical University of Athens, Greece
ISR1	Technion, Haifa, Israel
ISR2	Hebrew University of Jerusalem, Israel
KR	Advanced Inst. of Science and Tech., South Korea
MA1	ISP in Cambridge, MA
MA2	ISP in Cambridge, MA
MA3	ISP in Martha's Vineyard, MA
MA4	ISP in Massachusetts, MA
MD	ISP in Laurel, MD
MEX	National University of Mexico
MIT	Massachusetts Institute of Technology, MA
NC	ISP in Dhuram, NC
NL	Vrije University, Netherlands
NL2	ISP in Amsterdam, Netherlands
NYU	New York University, NY
NY	ISP in New York, NY
NZ	Victoria University of Wellington, New Zealand
SWD	Lulea University of Technology, Sweden
Swiss	Swiss Federal Institute of Technology
TW	National Taiwan University, Taiwan
UCSD	University of California, San Diego, CA
UK	ISP in London, UK
UT1	ISP in Salt Lake City, UT
UT2	ISP in Salt Lake City, UT

TABLE I
PARTICIPATING HOSTS AND THEIR LOCATIONS.

We run experiments over both TCP and UDP. When using TCP, every server keeps an active connection to every other server that it can communicate with, and periodically attempts to set up connections with servers to which it is not currently connected. We disable TCP's default waiting before sending small packets (Nagle algorithm, [27, Ch. 19]).

When using UDP, we implement failure detection using timeouts and acknowledgments. Like TCP, we use an exponentially weighted moving average (EWMA) to estimate the round trip time (RTT) to each host. Unlike TCP, we set the timeout to be twice the estimated RTT (whereas TCP sets it to be the RTT plus 4 times the mean deviation). When the timeout expires, the packet is retransmitted. We use exponential backoff as in TCP: each time the same packet is lost more than once, the timeout is doubled. However, we do not increase the timeout beyond 1 minute. Hosts test each other's liveness using heartbeats. We did not implement congestion control, since our experiments consume little bandwidth. We also do not order packet deliveries, which allows different invocations to succeed or fail independently without affecting each other's running times.

Our experiments consist of successive invocations of the primitive, called *sessions*. In each session, the initiator actually probes all the hosts, and logs the response time of every other host. Using an off-line analysis of this log data, we extrapolate the running times for two different quorum systems (majority and crumbling walls) and various probe sets; the running time is the time it takes until responses arrive from a quorum that is a subset of the chosen probe set. Sessions are initiated two minutes apart in order to allow messages sent in one session to arrive before the next session begins. This is especially important for TCP-based experiments, where messages are delivered in FIFO order. For the same reason, we limit the number of servers (initiators) that invoke sessions in a given experiment. We present our measured running times as cumulative distributions functions, that is, each curve shows the percentage of the sessions that terminate within x ms.

C. The Estimator

We devise a simple estimator for predicting running times based on link characteristics. Our estimator assumes that latency variations on different links are independent. We later use this estimator in order to investigate how accurately running times can be predicted based on end-to-end link characteristics. We begin by measuring the underlying TCP (or UDP) latency distributions during our experiments. Let rtt_{ah} be a random variable representing the round-trip latency over TCP between a host a and a host h. Consider a given initiator a, the majority quorum system with quorums of size of m, and a probe set P. Let $majority_a(P)$ denote the random variable capturing the time it takes a to hear from a majority after having probed the elements of P. Let $\mathcal{S}_k$ be the set of all subsets of P that are of size k. Then our estimator for the probability that an initiator a hears from at least m hosts within less than l units of time is as follows:

$$Pr[majority_a(P) < l] =$$

$$\sum_{i=m}^{n} \sum_{s \in \mathcal{S}_i} \prod_{h \in s} Pr[rtt_{ah} < l] \prod_{h \in P-s} (1 - Pr[rtt_{ah} < l])$$

IV. THE IMPACT OF PROBE SETS

In this section, we examine the relationship between probe set size and running time, for both majority and crumbling walls. We look at how this relationship is influenced by network dynamics (message loss rates, latency variation, and failures). The results presented in this section were gathered in a TCP-based experiment that lasted almost ten days and included 27 hosts. Not all hosts were up for the duration of the entire experiment. We show the results obtained at 4 of the hosts: in Taiwan (TW), in Korea (KR), at an ISP in Utah (UT2), and in Israel (ISR1). Each of these invoked a session once every two minutes on average, and in total, roughly 6700 times. Hosts also sent ping probes to each other once in two minutes.

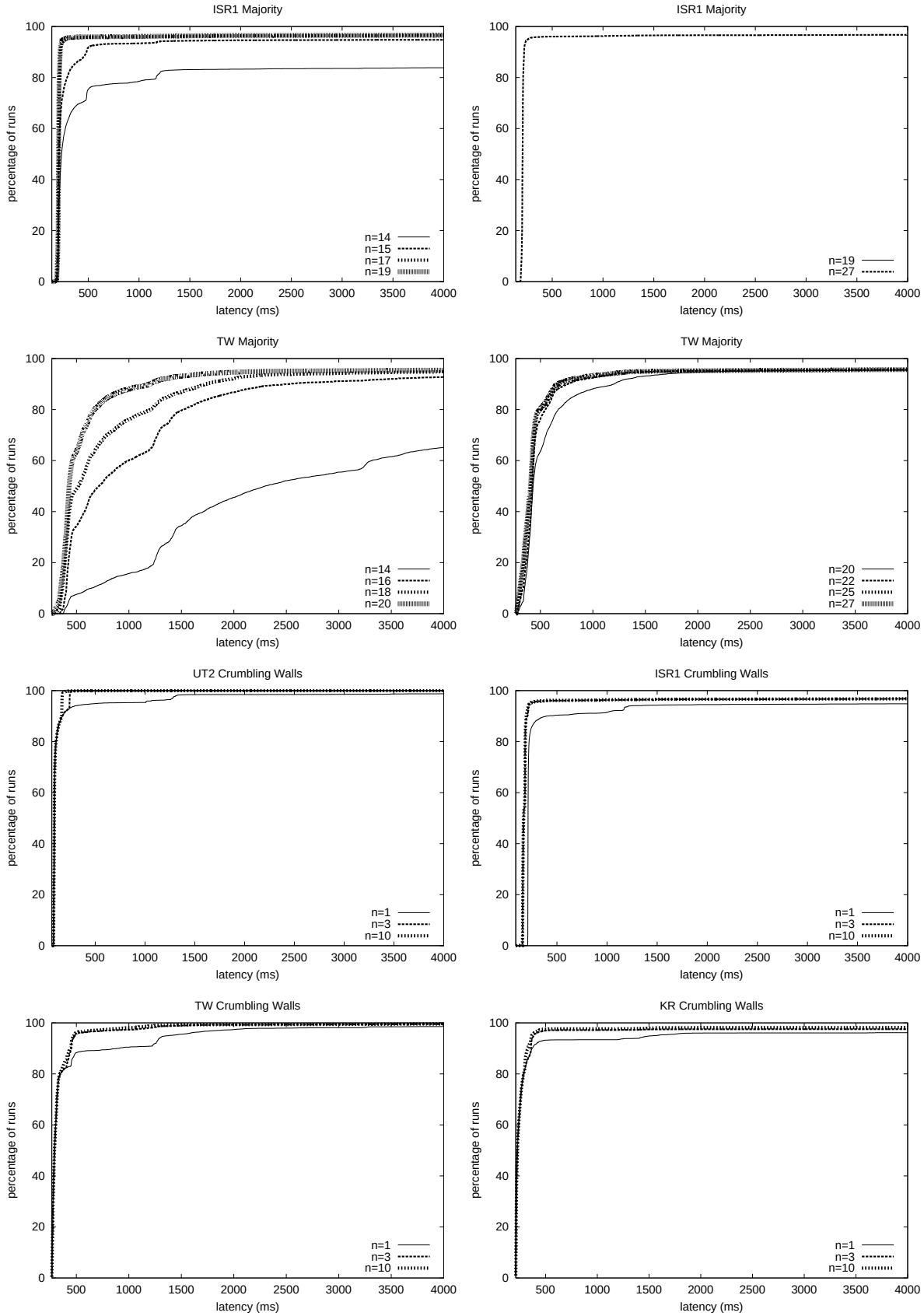

Fig. 2. Cumulative distribution of running times for different probe set sizes, quorum systems, and hosts. In majority, the probe set size is the number of hosts, and in crumbling walls, it is the number of rows.

A. The Majority Quorum System

Since we have a total of 27 hosts, a majority consists of at least 14 hosts (including the initiator). We look at performance improvements as the probe set size increases from 14 to 27. We chose the best probe set for every given size post-factum, based on link characteristics measured in the experiment. Specifically, for each session, we rank hosts according to the order in which they responded in that session (with the initiator ranked at 1). We then average the ranks over all sessions, and choose for the probe set of size k the hosts with the k best average ranks.

There are several arguments to be made for probe sets that are larger than the minimum. First, because of the dynamic nature of the Internet (changing routes, lost messages), the 14 hosts closest to the initiator do not remain the same for the entire duration of the experiment. Message loss, in particular, plays a significant role: a lost message from any of the 14 hosts in the minimal probe set, almost always increases the running time beyond the RTT of the 15th host. And no matter how reliable the links between the initiator and its closest 14 hosts are, they still have nonzero probabilities of dropping messages. Second, some hosts fail during the experiment. However, since failures during the experiment were infrequent, and network partitions were very short, the first factor plays a bigger role.

Our results indicate that all initiators other than TW can get very close to the best achievable running times with probe sets of 19 hosts. The most significant improvements are obtained when the probe set is increased to include 15 and then 16 hosts. The top two plots in Fig. 2 illustrate this observation. They show the cumulative distribution of running times in runs initiated at ISR1 for different probe set sizes. However, we observe a different phenomenon in TW. There, the performance continues to improve significantly as we increase the number of probed hosts up to 27, as illustrated in the second row of plots in Fig. 2.

To explain the different behavior in TW, we examine its link characteristics. Every initiator other than TW had highly reliable links with a low latency variance to most hosts. TW, on the other hand, had many links with highly variable latencies and loss rates of 25% or more (mostly to ISPs in North America). Table II shows the end-to-end characteristics as measured by ping from TW to other hosts. The *TCP connectivity* column indicates the percentage of the time that the TCP connection was up. We can see that hosts that have loss rates of 25% or more to TW also have the highest average latencies. At first glance, it would appear that probing these hosts is useless, since the high loss rate is compounded by the high latency. However, these links have the smallest minimum RTTs (shown in bold), which means that the best case involves responses from them. We also notice that the standard deviation is highest for these links, which means that low latency sessions are more probable. Therefore, the probability of getting good running times increases as we probe more of them.

We now examine how well our results can be predicted given our knowledge of the end-to-end characteristics. The first plot in Fig. 3 shows predicted cumulative running time distributions at TW, as computed by our estimator (cf. Section III-C); these estimates correspond to the measured values shown in the left plot on the second row of Fig. 2. The next five plots in Fig. 3 then examine the estimation error more closely. They reveal that for small probe sets, our estimator tends to under-estimate, whereas for larger probe sets, it over-estimates. With probe sets of size 16 and 17, the estimator under-estimates the low latency running times and over-estimates the high latency running times. These estimation errors are a consequence of the independence assumption. In reality, packet losses on different links from the same host are positively correlated. Low latencies are exhibited when no messages are lost, which occurs more often than predicted. The probability for multiple simultaneous losses (e.g., network partitions) is also higher than predicted, which causes less sessions than predicted to end within a given threshold (e.g., two seconds). Furthermore, the running time with a small probe set reflects the intersection of random variables, whereas with large probe sets, it reflects the union of many events. The probability of the intersection of events decreases when these events are independent as compared to when they are positively correlated. In contrast, the probability of a union of events increases when these events are independent as compared to events that are positively correlated. We have examined estimation errors for sessions initiated at TW as but one example; similar phenomena were observed with other initiators. The highest impact of correlated loss was observed on links with loss rates of up to 5%; on links with higher loss rates, loss was less correlated.

In Table III, we examine the impact that the probe set size has on availability. We compute availability as the percentage (rounded off to the closest integer) of sessions that successfully end with the response of a quorum within one minute. If no quorum responds within a minute, the session is considered to have failed. We observe that minimal probe sets achieve fairly low availability at all hosts, because the probability for one of the 14 hosts being down or unaccessible is not negligible. The availability greatly improves when the probe set size is increased to 15. With a probe set of 19 hosts, it generally reaches the availability of the complete probe set. Even with complete probe sets, some hosts do not achieve 100% availability due to network partitions.

	Majority				Crumbling Wall	
Host	min	15	19	complete	min	complete
MIT	92%	98%	100%	100%	99%	100%
AUS	68%	97%	100%	100%	98%	100%
UT2	76%	100%	100%	100%	99%	100%
ISR1	85%	96%	97%	97%	95%	97%
TW	87%	93%	96%	96%	99%	100%
KR	82%	96%	98%	99%	96%	98%

TABLE III
AVAILABILITY WITH DIFFERENT PROBE SETS.

Host	Loss Rate	Avg. Ping RTT	STD	Min. Ping RTT	TCP Connectivity	% TCP RTTs under 1 sec
TW	0%	0	0	0	100%	100%
UCSD	3%	232	25	**198**	100%	97%
Emulab	3%	238	26	216	100%	97%
NYU	3%	273	22	251	100%	97%
MIT	4%	303	398	256	100%	96%
CMU	4%	289	41	254	100%	96%
CU	3%	339	127	247	100%	97%
AUS	—	—	—	—	99%	96%
NL	3%	361	23	339	100%	96%
CA1	31%	482	626	**174**	95%	58%
NY	32%	445	853	234	96%	63%
SWD	3%	399	59	371	100%	96%
UT2	30%	743	1523	**171**	96%	58%
MA2	28%	742	1517	230	94%	59%
NC	32%	465	616	255	90%	63%
ISR2	3%	424	70	400	100%	97%
UT1	27%	979	1847	**189**	96%	55%
MA1	29%	645	712	238	96%	57%
ISR1	4%	551	2682	400	100%	94%
CA2	30%	606	1094	**179**	69%	45%
GR	3%	447	29	419	96%	93%
CND	35%	686	834	212	93%	51%
MA3	32%	774	1473	241	96%	54%
NZ	35%	636	830	271	91%	43%
KR	11%	357	163	200	42%	40%
CA3	32%	1047	2091	**178**	15%	10%
Swiss	3%	384	22	362	15%	15%

TABLE II

END-TO-END LINK CHARACTERISTICS FROM TW TO OTHER HOSTS DURING THE EXPERIMENT.

B. Crumbling Walls

We now analyze the crumbling wall quorum system shown in Fig. 4. Its construction is based on Section 4.1 of [10]. It consists of 10 rows, varying in width. The hosts were placed in rows as follows:

- The first (bottom) row includes 4 hosts at North American universities, which were up for the entire experiment. This improves performance for hosts in North America, which constitute a majority of the hosts.
- Our ping traces indicate that hosts located in Europe and Israel are connected to each other by good links (i.e., links with low latencies, latency variations, and loss rates). In order to improve performance for these hosts, we placed such hosts in the second row.
- Rows 3–5 include other North American hosts that did not crash, as well as ISR2.
- Swiss was under firewall restriction for a portion of the experiment, and was therefore placed at the top.
- The remaining hosts occupy the remaining rows.

We look at the performance improvements obtained as we increase the number of rows in the probe set from 1 to 10. The plots in the last two rows in Fig. 2 show the running time distributions for four different initiators and various probe set sizes. Depending on where the initiators are located, they see different gains. As expected, the North American host UT2 is affected very little by the addition of rows, because the hosts

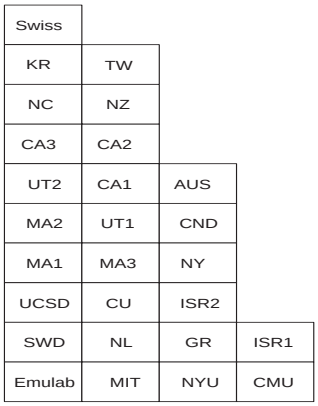

Fig. 4. Our crumbling wall quorum system.

of the first row are in North America. TW, which has lossy links to many hosts, and ISR1, which has good connectivity to the second row hosts, see bigger gains. However, we note that even at these hosts, the performance gains from probing two additional rows in the crumbling wall are still smaller than those obtained by probing an additional (15th) host beyond the minimal probe set in the majority system. This is due to the fact that the crumbling wall requires accessing much fewer hosts, which decreases the probability of having at least one host in the first quorum fail.

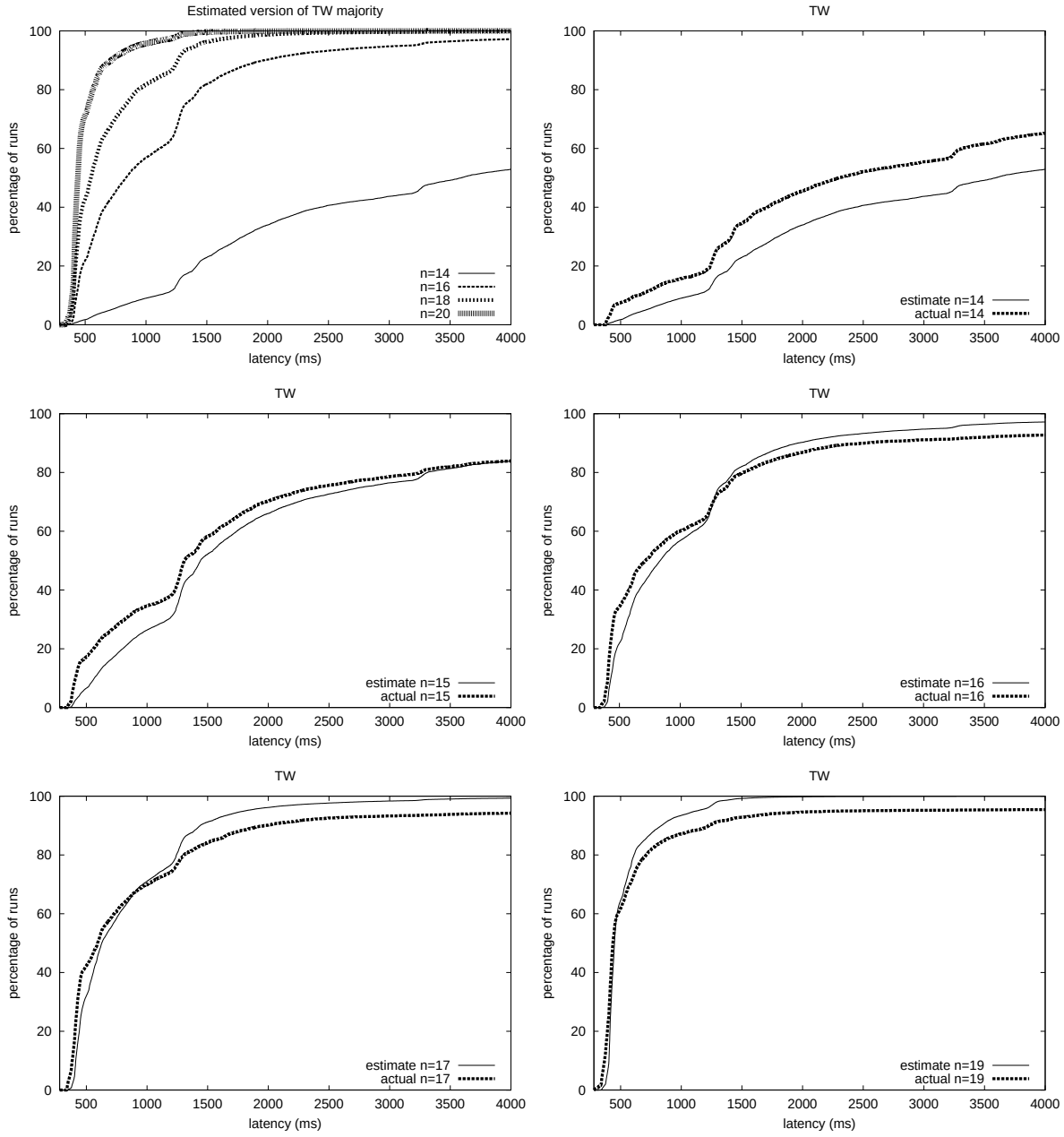

Fig. 3. Estimated vs. actual running time distributions at TW.

Table III shows that unlike with majority, with the crumbling wall, the minimum probe sets already achieve fairly high availability. Furthermore, the maximum availability of the crumbling wall is generally as good as, and in one case even better than, that of majority.

V. Communicating over UDP

In this section, we compare the results from running over the two most popular Internet transport protocols: UDP and TCP. We experiment with 26 hosts, during a period of four and a half days. Each host concurrently runs both the TCP and UDP servers. Each server invokes a session of each protocol independently every two minutes on average. A total of roughly 3100 sessions of each protocol were invoked by each initiator during the experiment.

We show the results measured at two hosts, located at MIT and Taiwan (TW), using the majority quorum system. Since we use 26 hosts, a quorum consists of at least 14. We measured the underlying link characteristics using our UDP traces (rather than ping). Tables IV and V show the end-to-end characteristics measured from TW and MIT (respectively) to other hosts that were up for the entire duration of the experiment. The *unidirectional loss%* column in Table IV shows the loss rate for messages that travel in one direction

only (to TW), whereas, the column labeled *bidirectional loss%* in the same table shows the loss rate for messages traveling the entire round trip.

Host	RTT (ms)	unidirectional loss%	bidirectional loss%
CA2	170	1.6%	4.56%
NC	259	1.53%	3.46%
CA5	159	0.16%	3.26%
CND	212	0.43%	3.16%
MD	232	0.5%	3.13%
CHI	216	0.2%	3%
CA4	143	0.3%	2.8%
UT1	226	0.43%	2.76%
NY	231	0.33%	2.73%
NL2	315	0.26%	2.7%
CA1	237	0.93%	2.4%
UCSD	206	0.93%	2.43%
UK	305	0.3%	2.2%
CA3	178	0.3%	2.16%
CA6	160	0.3%	1.9%
GR	356	0.46%	0.56%
MIT	221	0.1%	0.5%
ISR1	371	0.1%	0.3%
Emulab	226	0.1%	0.13%
CMU	226	0.1%	0.13%

TABLE IV
LINK CHARACTERISTICS FROM TW TO HOSTS THAT DID NOT CRASH DURING THE EXPERIMENT.

We first compare the protocols in terms of their optimal probe sets. Fig. 5 shows the cumulative running time distributions for different majority probe set sizes for both TCP and UDP. At MIT, for both UDP and TCP, the minimal running time is achieved with a majority of 15 hosts. For TW, the optimal running time in both cases is achieved with a probe set of size 26. However, the performance gains from increasing the probe set from 14 to 18 are greater with TCP than with UDP. Tables IV and V help explain why TW requires a much larger probe set than MIT in order to achieve the best running time: they show that MIT has only two links with loss rates exceeding 0.5%, whereas TW has 14 links with loss rates exceeding 2%.

The next two rows in Fig. 5 illustrate how the underlying protocol (TCP versus UDP) affects the running time for a given probe set. We notice that where loss rates do not play a significant role, the curves are virtually identical. This is the case with MIT, regardless of the size of the probe set, due to its very low loss rates. This is also the case for large probe sets in TW, because with such sets, there are sufficiently many alternative links to be able to mask losses. However, when there are no alternatives to lossy links, e.g., TW with probe sets of 14 and 15, UDP outperforms TCP, due to the less conservative retransmission timeouts we implemented with UDP.

We now use our UDP traces in order to study loss correlation. Specifically, we examine how the loss probability of a message sent to a particular host changes if we know that messages sent to other hosts in the same session were lost. This is illustrated in Fig. 6: the top (solid) curve in Fig. 6 shows the conditional probability that a message sent from TW to the UK is lost, given that at least x messages sent to other hosts in the same session are lost. When $x = 0$, this is simply the loss rate on the link. The second (dashed) curve plots the conditional loss probabilities for messages sent from TW to CMU. The figure clearly shows that loss rates are highly correlated. Given that at least two messages in a given session were lost, the loss rate to the UK goes up from 2.2% to 25%, and when at least 6 other messages are lost, the loss rate is already 100%. This explains the inaccuracy of the estimator observed in the previous section.

Host	RTT (ms)	STD	bidirectional loss%
CA2	86	43	1.46%
NC	64	215	0.36%
CA5	86	208	0.1%
CND	44	42	0.06%
MD	17	42	0.23%
CHI	37	41	0.16%
CA4	79	41	0.16%
UT1	67	41	0.1%
NY	22	112	0.06%
NL2	104	41	0.1%
CA1	237	117	2.4%
UCSD	94	208	0.5%
UK	90	41	0.06%
CA3	116	216	0.06%
CA6	86	43	0.06%
GR	143	129	0.56%
TW	226	191	0.2%
ISR1	189	203	0.33%
Emulab	69	208	0.06%
CMU	22	7	0.03%

TABLE V
LINK CHARACTERISTICS FROM MIT TO HOSTS THAT DID NOT CRASH DURING THE EXPERIMENT.

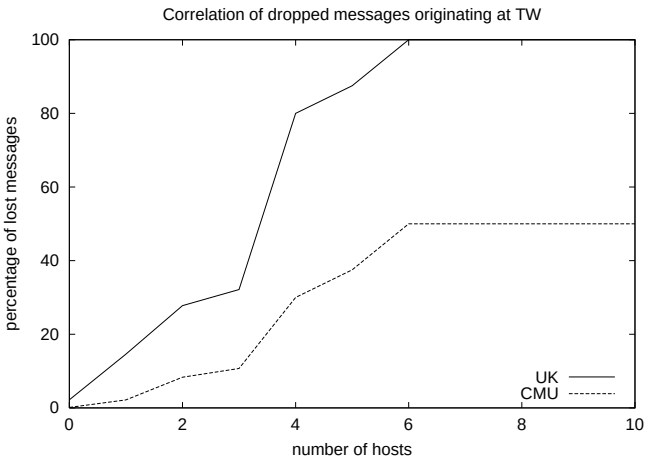

Fig. 6. Conditional probabilities of message loss from TW.

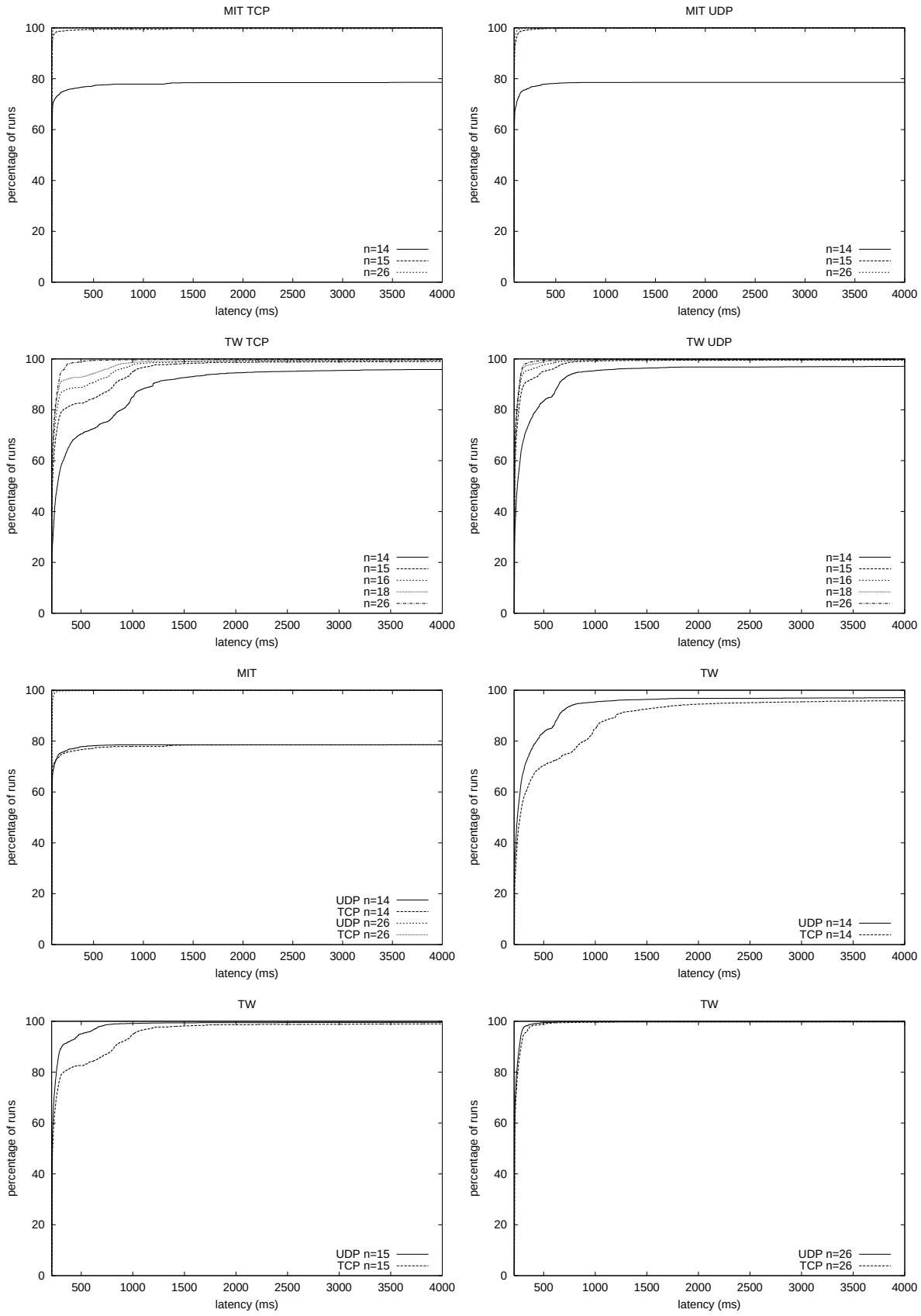

Fig. 5. Cumulative running time distributions using majority and various probe sets, TCP versus UDP.

In order to understand correlated loss better, we examined the traceroute data gathered during the previous experiment. We have observed that outgoing links from TW to many other hosts traverse the same routers, whereas links incoming into TW traverse different paths. We therefore hypothesized that most of the correlated losses occur on these shared outgoing paths. Using the UDP traces gathered during this experiment, we now examine the difference between the unidirectional and bidirectional loss rates involving TW. Table IV reveals that indeed, the loss rates are not symmetric. In some cases (rows 3–6 in Table IV) the probability of losing a message headed to TW is less than 0.5%, whereas the bidirectional loss rate on the same link exceeds 3%. This means that most of the losses occur on packets traveling away from TW. It has been previously shown (e.g., in [13]) that a large portion of the Internet paths (routes) are asymmetric. Our results show that such routing asymmetries can result in large discrepancies in unidirectional loss rates.

VI. Conclusions

We have studied the performance of quorum-based replication over the Internet. We examined the impact that the probe set has on system running time and availability. We first looked at the majority quorum system. Our study has shown that majority replication does not perform well with small probe sets. However, a moderate increase in the probe set size generally yields high performance gains. For example, probing 15 hosts instead of 14 out of 27 greatly reduces the running time at most hosts. After adding a few hosts to the probe set, however, a point is reached where adding more hosts is no longer beneficial. We further observed that the availability of majority with minimal probe sets is unsatisfactory, but increasing the probe set by a few hosts again achieves high availability.

We then looked at a crumbling wall quorum system, and showed that it is preferable to majority. Its performance is greatly superior to that of majority with minimal probe sets, and it achieves better performance than majority even with complete probe sets. Furthermore, although classical availability analysis [11] suggests that majority is the most available quorum system, we observe that in reality, majority is generally not more available than the crumbling wall even with complete probe sets. This is because the classical analysis assumes IID failures and full connectivity, whereas in practice, correlated failures and network partitions play a major role (a similar observation was made in [12]). When minimal probe sets are used, the availability of crumbling walls is significantly superior to that of majority.

Our study has focused on running time and availability, and ignored other factors such as load and load balancing. Note that the overall system load is much smaller with crumbling walls than with majority, since crumbling wall quorums and probe sets are smaller. However, since in our approach each initiator chooses the probe set that optimizes its local performance, we get that the hosts placed in the first row of the crumbling wall, which all have good connectivity, shoulder all the load. Devising probe sets that achieve load balancing in addition to performance and reliability is an interesting direction for future work.

We have conducted our study by running experiments on thirty hosts widely distributed over the Internet. We have shown that running the system on the actual network is important, because high cross-correlation among loss probabilities of messages sent to different hosts in one session render analysis based on independent per-link end-to-end traces inaccurate.

Acknowledgments

The majority of the hosts we used belong to the RON Project [16], which is funded by DARPA. We are grateful to Dave Andersen for his dedicated assistance with these machines. We are also indebted to Sebastien Baehni, Danny Bickson, Danny Dolev, Alan Fekete, Rachid Guerraoui, Yuh-Jzer Joung, Mark Moir, Sergio Rajsbaum, Greg Ryan, and Ilya Shanayderman for hosting our experiments on their machines and for providing technical support.

References

[1] R. Thomas, "A Majority Consensus Approach to Concurrency Control for Multiple Copy Databases," *ACM Trans. on Database Systems*, vol. 4, no. 2, pp. 180–209, June 1979.

[2] D. K. Gifford, "Weighted voting for replicated data," in *ACM SIGOPS Symposium on Operating Systems Principles (SOSP)*, December 1979.

[3] C. A. Thekkath, T. Mann, and E. K. Lee, "Frangipani: A scalable distributed file system," in *ACM SIGOPS Symposium on Operating Systems Principles (SOSP)*, 1997, pp. 224–237.

[4] D. Skeen, "A quorum-based commit protocol," in *6th Berkeley Workshop on Distributed Data Management and Computer Networks*, Feb 1982, pp. 69–80.

[5] A. El Abbadi, D. Skeen, and F. Christian, "An efficient fault-tolerant algorithm for replicated data management," in *ACM SIGACT-SIGMOD Symposium on Principles of Database Systems (PODS)*, March 1985, pp. 215–229.

[6] M. Herlihy, "A quorum-consensus replication method for abstract data types," vol. 4, no. 1, pp. 32–53, Feb. 1986.

[7] D. Malkhi and M. K. Reiter, "An architecture for survivable coordination in large-scale systems," *IEEE Transactions on Knowledge and Data Engineering*, vol. 12, no. 2, pp. 187–202, March/April 2000.

[8] J.-P. Martin, L. Alvisi, and M. Dahlin, "Minimal byzantine storage," in *16th International Symposium on DIStributed Computing (DISC)*, October 2002.

[9] J. Yin, J. Martin, A. Venkataramani, L. Alvisi, and M. Dahlin, "Separating agreement from execution for byzantine fault tolerant services," in *19th ACM SIGOPS Symposium on Operating Systems Principles (SOSP)*, Oct. 2003.

[10] D. Peleg and A. Wool, "Crumbling walls: A class of practical and efficient quorum systems," *Distributed Computing*, vol. 10, no. 2, pp. 87–98, 1997.

[11] ——, "Availability of quorum systems," *Inform. Comput.*, vol. 123, no. 2, pp. 210–223, 1995.

[12] Y. Amir and A. Wool, "Evaluating quorum systems over the Internet," in *IEEE Fault-Tolerant Computing Symposium (FTCS)*, June 1996, pp. 26–35.

[13] V. Paxson, "End-to-end routing behavior in the Internet," in *ACM SIGCOMM*, 1996, pp. 25–38.

[14] ——, "End-to-end Internet packet dynamics," in *ACM SIGCOMM*, September 1997.

[15] Y. Zhang, N. Duffield, V. Paxson, and S. Shenker, "On the constancy of Internet path properties," in *ACM SIGCOMM Internet Measurement Workshop*, November 2001.

[16] D. G. Andersen, H. Balakrishnan, F. Kaashoek, and R. Morris, "Resilient overlay networks," in *SOSP*. ACM, Oct. 2001, pp. 131–145.

[17] S. Savage, A. Collins, E. Hoffman, J. Snell, and T. Anderson, "The end-to-end effects of Internet path selection," in *ACM SIGCOMM*, September 1999, pp. 289–299.

[18] M. Dahlin, B. Chandra, L. Gao, and A. Nayate, "End-to-end WAN service availability," *ACM/IEEE Transactions on Networking*, vol. 11, no. 2, Apr. 2003.
[19] S. Saroiu, K. Gummadi, and S. Gribble, "A measurement study of peer-to-peer file sharing systems," in *Multimedia Computing and Networking*, January 2002.
[20] R. Bhagwan, S. Savage, and G. Voelker, "Understanding availability," in *2nd International Workshop on Peer-to-Peer Systems (IPTPS'03)*, Feb. 2003. [Online]. Available: http://www-cse.ucsd.edu/users/voelker/pubs/avail-iptps03.pdf
[21] M. Naor and A. Wool, "The load, capacity, and availability of quorum systems," vol. 27, no. 2, pp. 423–447, 1998.
[22] D. Peleg and A. Wool, "How to be an efficient snoop, or the probe complexity of quorum systems," *SIAM Journal on Discrete Mathematics*, vol. 15, no. 3, pp. 416–433, 2002.
[23] O. Bakr and I. Keidar, "Evaluating the running time of a communication round over the Internet," in *ACM Symposium on Principles of Distributed Computing (PODC)*, July 2002, pp. 243–252.
[24] T. Anker, D. Dolev, G. Greenman, and I. Shnayderman, "Evaluating total order algorithms in wan," in *SRDS Workshop on Large-Scale Group Communication*, 2003.
[25] O. Bakr, "Performance evaluation of distributed algorithms over the Internet," Master's thesis, MIT Department of Electrical Engineering and Computer Science, Feb. 2003, Master of Engineering.
[26] R. Jimenez-Peris, M. Patino-Martinez, G. Alonso, and B. Kemme, "How to Select a Replication Protocol According to Scalability, Availability, and Communication Overhead," in *IEEE International Symposium on Reliable Distributed Systems (SRDS)*. New Orleans, Louisiana: IEEE CS Press, Oct. 2001. [Online]. Available: citeseer.nj.nec.com/jimenez-peris01how.html
[27] R. Stevens, *TCP/IP Illustrated*. Addison-Wesley, 1994, vol. 1.