1

SIMULATION MODELING OF WEAK-CONSISTENCY PROTOCOLS

Richard A. Golding and Darrell D. E. Long

Computer and Information Sciences Board
University of California
Santa Cruz, California 95064

ABSTRACT

Weak-consistency replication protocols can be used to build wide-area services that are scalable, fault-tolerant, and useful for mobile computer systems. We have developed the *timestamped anti-entropy* protocol. In it, pairs of replicas periodically exchange update messages; in this way updates eventually propagate to all replicas. We present results from a detailed simulation analysis of the fault tolerance and the consistency provided by this protocol. The protocol is extremely robust in the face of site and network failure, and it scales well to large numbers of replicas.

1 INTRODUCTION

We have developed an architecture for building wide-area distributed services that emphasizes scalability and fault tolerance. This allows applications to respond gracefully to changes in demand and to site and network failure. It uses a single mechanism – weak-consistency group communication – to support both wide-area services and mobile computing systems.

The architecture uses data *replication* to meet availability demands and enable scalability. A group of servers or replicas cooperate to form a service. The replication is dynamic in that new servers can be added or removed to accommodate changes in demand. The system is asynchronous, and servers are as independent as possible; it never requires synchronous cooperation of large numbers of sites. This improves its ability to handle both communication and site failures. It uses large numbers of replicas, so replicas can be placed near

clients and to spread query load over many sites.

We use *group communication* as the structuring principle for the architecture. A group consists of a set of *principals,* or group members, which communicate among themselves to coordinate a replicated service. The group members act as servers in a replicated service. They follow a *group communication protocol* when doing so, which provides a multicast service from one group member to the entire group. Clients can also multicast messages to the group, and the underlying protocol ensures that the communication reaches the appropriate group members.

Existing strong-consistency group communication protocols only work well with small groups. As the number of members increases, availability and response time decrease beyond the level acceptable for interactive applications, because they require the synchronous cooperation of a large number of principals. Instead we have turned our attention to *weak-consistency* protocols.

When weak-consistency group communication protocols are used, updates are first delivered to one site, then propagated asynchronously to others. The value a server returns to a client read request depends on whether that server has observed the update yet. Eventually, every server will observe the update. This allows updates to be sent using bulk transfer protocols, which provide the best efficiency on high-bandwidth high-latency networks. It also allows the group communication protocol to mask many failures, obviating the need for explicit recovery protocols in many cases. Several information systems, such as the Xerox Clearinghouse system [4], have used similar techniques.

We have developed the *timestamped anti-entropy (TSAE)* weak-consistency protocol. It builds on our work on frameworks for implementing and analyzing group communication protocols [8]. This protocol provides *reliable, eventual* message delivery, ensures that message logs can be purged reliably, and supports mobile computer systems.

Analytical modeling of replication and group communication protocols is generally difficult, particularly when many principals are involved. We, like many other researchers, adopted simulation – both discrete event and Monte Carlo – as the appropriate method for evaluating our wide-area architecture.

In the remainder of this section, we will justify why we believe weak consistency protocols to be necessary for wide area systems, and discuss the measures we used to evaluate the TSAE protocol. Next we present the protocol. We then present the details of its analysis, concentrating on message reliability, delivery

latency, message traffic, and inconsistency among principals.

The wide-area network environment

Existing consistent replication protocols are unsuited for wide-area applications because of the latency, failure, and scale of those networks. Latency affects the response time of the application, and can vary from a few milliseconds, for two hosts connected by an Ethernet, to several hundred milliseconds for hosts on different continents communicating through the Internet. Packet loss rates often reach 40%, and can go higher [6]. The Internet has many points that, on failure, partition the network, and at any given time it is usually partitioned into several non-communicating networks. Further, in 1992 the Internet included more than a million hosts [12], with a potential user base several times larger. This has led to query loads on some services exceeding the capacity of a single workstation and the network links that support it [5]. Any group communication protocol that requires interactive communication with many principals will not work in this environment.

The introduction of mobile computers exacerbates this problem further. These systems spend most of their time disconnected from other systems, or perhaps connected by an expensive or low-bandwidth radio link. Services that are to support such systems must allow clients that are disconnected to continue to operate, without communicating with outside servers. This can be accomplished by placing a group member on the mobile system and allowing the copy to diverge from the "correct" value. Weak consistency protocols ensure that this divergence can be reconciled when the mobile system is reconnected to the network. The timestamped anti-entropy protocol allows mobile systems with limited bandwidth to measure how far they have diverged by exchanging a small summary of the state of the group member with another member.

Despite these restrictions, users expect a service to behave as if it were provided on a local system. The response time of a wide-area application should not be much longer than that of a local one. Further, users expect to use the service as long as their local systems are functioning.

We assume that principals have access to pseudo-stable storage such as magnetic disk that will not be affected by a system crash. Principals, or the hosts on which they run, have loosely synchronized clocks. Principals and hosts fail by crashing; that is, when they fail they do not send invalid messages to other principals and they do not corrupt stable storage. Hosts can temporarily fail and then recover. Principals have two failure modes: temporary recoverable fail-

ures, and permanent removal from service. The network is sufficiently reliable that any two principals can eventually exchange messages, but it need never be free of partitions. *Semi-partitions* are possible, where only a low-bandwidth connection is available.

We believe that weak-consistency protocols provide a good solution for building a replicated service on a wide-area network. Clients can communicate with the replica nearest them, reducing message traffic on the network and spreading query load over many servers. The TSAE protocol, in particular, scales well to large numbers of replicas. The protocols make many faults, such as network partitions and system crashes, invisible to the replication service by allowing copies to diverge, then reconciling them when the fault is repaired. This is a natural way of handling mobile, disconnected computer systems as well.

Measures

How well can weak-consistency protocols be expected to work? How do they compare to other approaches? We used several different performance measures to answer these questions.

Two of the measures concern individual messages. *Message reliability* measures how often the protocol will fail to deliver a message because principals are removed from service without notice, while *message latency* measures how long the protocol takes to deliver a message.

The other two measures aggregate the effects of many messages. *Message traffic* indicates how many messages applications using weak-consistency protocols will generate, and indirectly how they interfere with other network activity. *Consistency* measures how up-to-date each principal can be expected to be.

2 PROTOCOL DESCRIPTION

Replicated data can be implemented as a group of replica principals that communicate through a *group communication protocol*. The group communication protocol generally provides a *multicast* service that sends a message from one principal to all other principals in the group.

The protocol determines the *consistency* of each principal by controlling the order messages are sent among them. Consistency is often defined in terms of the value that each principal holds. However, that value is determined by the

messages that the principal has received, so we define consistency in terms of messages. *Weak* consistency protocols guarantee that messages are delivered to all members but do not guarantee when.

Kinds of consistency

The consistency provided by a group communication protocol can be classified by the guarantees it provides. This includes guarantees on *message delivery, time of delivery,* and *delivery ordering.* In general, strong guarantees require multiphase synchronous protocols while weaker guarantees allow efficient asynchronous protocols.

Messages can either be delivered *reliably* or not. We have identified four levels of reliability: *atomic,* where the message is either delivered to every member, or to none; *reliable,* where the message is delivered to every functioning principal, or to none; *quorum,* where the message is delivered to at least some fraction of the membership; and *best-effort,* where delivery is attempted but not guaranteed. These levels differ in the way they handle failed principals and unreliable communication media.

Likewise, the communication protocol can deliver messages *synchronously,* within a *bounded* time, or *eventually* in a finite but unbounded time. This guarantee is orthogonal to reliability.

Messages will be delivered to principals in some order, perhaps different from the order in which they are received. There are several possible orders, including total, causal [11, 10], and bound inconsistency [13, 3]. Weaker orderings include a *per-principal* or *FIFO channel* ordering, where the messages from any particular principal are delivered in order, but the streams of messages from different principals may be interleaved arbitrarily.

Weak consistency protocols provide reliable delivery, and can provide one of several delivery orderings, but only guarantee eventual message delivery. In particular, all principals are guaranteed to agree in finite but unbounded time if no further messages are sent.

These protocols are generally implemented using background message propagation. When a client wishes to update the value a group maintains, it sends a message to one group member. This member then forwards it to another member. Later both members forward it to other members, and so on until it has spread throughout the group.

Anti-entropy protocols use this technique, and guarantee reliable delivery. Principals using this protocol periodically exchange messages with other principals, potentially continuing this process forever. Other weak consistency protocols, such as rumor mongery [4], do not guarantee reliable delivery, because principals use probabilistic rules to determine when to stop propagating a message.

Timestamped anti-entropy

We have developed a new group communication protocol called *timestamped anti-entropy* [7]. It provides reliable, eventual delivery, so that messages are delivered to every principal in the group even if some have temporarily failed or are disconnected from the network. We have also developed a related group membership mechanism that provides for adding and removing members from the group [9]. The trade-offs are that the protocol may have to delay message delivery until a fault is repaired, that members must maintain logs on disk that are not compromised by failure and recovery, and that timestamp information must be appended to every message.

We have developed the TSAE protocol, a new group communication protocol that provides reliable, eventual delivery [7]. Like other weak-consistency protocols, update messages originate at a single principal and are propagated in the background to others. Unlike other protocols, TSAE supports several message delivery orders, mobile computer systems, and provably correct purging of message logs. We have also developed a related group membership mechanism that handles adding and removing members from the group [9].

When a principal wishes to send a message, presumably as the result of a client performing an update operation, the message is marked with a timestamp and the identity of the sending principal, and written to a message log. This log is maintained on stable storage, so that it survives temporary crashes. It is organized as a set of sub-logs, one for each principal in the group.

Each principal maintains two *timestamp vectors* on stable storage in addition to the message log. Each vector is indexed by principal identifiers. The summary vector maintains a summary of the messages in the log, and when it holds a timestamp value for a particular principal, all messages sent by that principal with lesser or equal timestamp have been received. The entry in the summary vector for principal P is always larger than the greatest timestamp recorded in the message sub-log for P.

The acknowledgment vector summarizes the messages that have been received

by other principals. Since clocks are loosely synchronized, a principal can acknowledge that it has received all messages sent by any principal up to a particular time.[1]

From time to time, a principal will select another principal as its partner, and the two will exchange messages in an *anti-entropy session.* The session begins with the principals exchanging their summary vectors. Using this information, each principal can determine the messages in its log that its partner has not yet received. These messages are sent using a reliable stream communication protocol. The session ends with the principals exchanging their acknowledgment vectors.

When a session is complete, both principals have received exactly the same set of messages. Moreover, they have received a continuous sequence of messages from each principal. To see that this is so, consider the first time one principal, call it A, contacts another principal, B. Each principal will only have messages it has sent in its log, and its summary vector will contain zero timestamps except for its own entry. A and B will exchange messages, each receiving a continuous sequence of messages from the other. Some time later, A will contact B again and receive more messages sent by B. At the beginning of the session, the summary timestamp that A records for B will be earlier than the summary timestamp that B records for itself. B will forward all the messages that A has not yet received, and A will once again have a continuous sequence of messages from B in its log.

The acknowledgment vector is used to keep the log from growing without bound. At the end of an anti-entropy session, principal A finds the minimum timestamp, min, in its summary vector. A has received every message from any sender that has a lesser or equal timestamp. A implicitly acknowledges all these messages by setting its entry in its acknowledgment vector, acknowledgment$[A]$, to min. A log entry can be purged when every other process has observed it, which is true when the minimum timestamp in the acknowledgment vector is greater than the timestamp on the log entry.

There are several variations on the basic TSAE protocol. The timestamp vectors and message timestamps can be used to order messages before they are delivered from the log. Anti-entropy sessions can be augmented by a best-effort multicast to speed propagation, and principals can use various policies to select partners.

[1] We have also developed a similar protocol that requires $O(n^2)$ state per principal rather than $O(n)$, but allows unsynchronized clocks. This alternate protocol was discovered independently by Agrawal and Malpani [1].

Table 1 Partner selection policies.

Random policies:	
Uniform	Every neighbor principal has an equal probability of being randomly selected.
Distance-biased	Nearby neighbors have a greater probability than more distant neighbors of being randomly selected. Should reduce traffic on long-distance backbone links.
Oldest-biased	The probability of selecting a neighbor is proportional to the age of its entry in the summary vector.
Deterministic policies:	
Oldest-first	Always selects the neighbor n with the oldest value in the summary vector.
Latin squares	Builds a deterministic schedule guaranteed to propagate messages in $\Theta(\log n)$ rounds [2].
Topological policies:	
Ring	Organizes the principals into a ring.
Binary tree	Principals are organized into a binary tree, and propagate messages randomly along the arcs in the tree.
Mesh	Organizes the principals into a two-dimensional rectangular mesh.

Best-effort multicast combined with anti-entropy will spread information rapidly. When a principal originates a message, it can multicast it to other principals. Some of them will not receive the multicast, either because the network dropped the message or because the principal was temporarily unavailable. These members will receive the message later when they conduct an anti-entropy session with another member that has received the message. This speeds dissemination when message delivery order is not important.

Partner selection

There are several possible policies for selecting a partner for an anti-entropy session. Table 1 lists eight of them. The TSAE protocol requires only that every neighbor eventually be contacted to ensure that messages are delivered reliably, and weaker constraints can work for some network topologies.

The policies can be divided into three classes: random, deterministic, and topological. Random policies assign a probability to each neighbor, then randomly select a partner for each session. The deterministic policies use a fixed rule to determine the neighbor to select as partner, possibly using some extra state such as a sequence counter. Topological policies organize the principals into some fixed graph structure such as a ring or a mesh, and propagate messages along edges in the graph.

3 MESSAGE RELIABILITY

The timestamped anti-entropy message delivery protocol provides reliable eventual delivery. However, reliable delivery does not guarantee that a message is delivered when its sender fails. For the TSAE protocol to fail to deliver a message, every principal that has received a copy – which includes the sender – must fail. This section examines how often this happens.

Delivery reliability can be measured by the probability that a message will be delivered to every principal before all recipients can fail. The probability is affected by the rate at which anti-entropy sessions propagate messages, and the rate at which principals fail.

In practice, the only way a principal can fail is in a sudden, catastrophic removal from service – a fire or hardware failure, for example. This sort of failure is extremely rare for systems on the Internet. The analysis in this section, however, explores a wide range of failure rates.

Monte Carlo simulation

Message loss can be modeled using a state transition system like that shown in Figure 1. Each state is labeled with a pair $\langle m, f \rangle$, where m is the number of functioning principals that have observed a message, and f is the total number of functioning principals. The system starts in state $\langle 1, n \rangle$, with one principal having observed a message out of n possible (5 in the example). The system can then either propagate the information using anti-entropy, in which case the system moves to state $\langle 2, n \rangle$, or a principal can be removed from service and the system moves into state $\langle 0, n-1 \rangle$. The message has been lost when the system reaches a state $\langle 0, x \rangle$, and it is delivered when it reaches $\langle x, x \rangle$.

Anti-entropy and principal failure are treated as Poisson processes with rate λ_a and λ_f, because Poisson processes are easy to model and analyze. Real systems often follow more complex distributions, but this Poisson process modeling are known to be robust even if the underlying distribution is not exponential.

The rate of *useful* anti-entropy sessions, where a principal that has received the message contacts one that has not, is a function of m, f, and the partner selection policy. In particular, f principals will be initiating anti-entropy sessions. If principals choose their partners randomly, each principal that has observed the update has a chance $(f-m)/(f-1)$ of contacting a principal that has not yet observed the update. Since anti-entropy is a Poisson process, the rate of

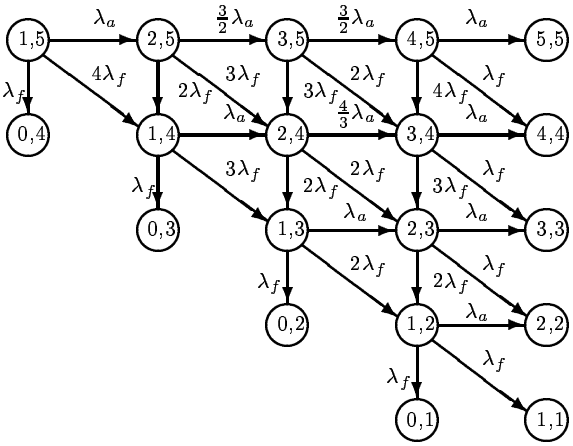

Figure 1 Model of message receipt and failure for five principals. This model only includes permanent failure; transient failure and recovery would add an additional dimension of states.

useful anti-entropy sessions is

$$m\frac{f-m}{f-1}\lambda_a.$$

Since removal from service is a permanent event, the state transition graph is acyclic, with $\Theta(n^2)$ states in the number of principals. The probability p_i of reaching each state i can be computed using a sequential walk of the states. The probability density functions $p_i(t)$ of the time at which the system enters each state can be derived analytically or numerically. The analytic solution for $p_i(t)$ can be found by convolving the entry-time distribution $p_j(t)$ for each predecessor state j with the probability density of the time required for the transition from j to i. Alternately, the system can be solved numerically using a simple Monte Carlo evaluation.

Results

Figure 2 shows the probability of removal from service interfering with message delivery for different numbers of principals. The probability is a function of $\rho = \lambda_a/\lambda_f$, the ratio of the anti-entropy rate to the permanent site failure rate. The two graph is plotted using a linear vertical scale, which emphasizes the behavior for small values of ρ. The probability asymptotically approaches zero as ρ increases.

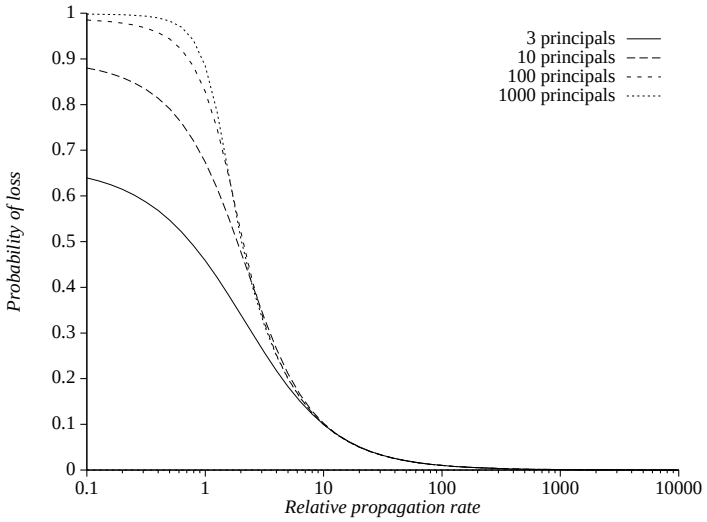

Figure 2 Probability of failing to deliver a message to all sites (linear vertical scale). The relative propagation rate ρ is the ratio of the anti-entropy rate λ_a to the permanent site failure rate λ_f. The linear scale emphasizes the effect of the number of principals for small values of ρ.

Internet sites generally are removed from service after several years of service, and then usually with enough notice to run a leave protocol. The anti-entropy rate is therefore likely to be many thousands of times higher than the permanent failure rate. As a result, there will be almost no messages lost because of removal from service.

Volatile storage

Some implementations may buffer messages in volatile storage before copying them to the stable message log. Write-back caches are used by many operating systems. These implementations will lose the information in volatile storage when a principal temporarily fails and recovers.

Volatile storage complicates the state transition model. States must be labeled with four values: the number of functioning principals that have not observed a message, the number that have written it to volatile store, the number that have written it to disk, and the number that have temporarily failed. The state transitions are complex and the solution is impractical for realistic numbers of principals.

However, the effect of volatile storage can be bounded by considering the probability that a failure will occur while there are messages that have not been made stable. Assume that temporary failure is a Poisson process with rate λ_t and that volatile data is flushed to stable storage every s time units. The probability that a failure occurs before writeback is

$$p = \frac{-2e^{-s\lambda_t} + s^2\lambda_t^2 - 2s\lambda_t + 2}{2s\lambda_t^2}.$$

For a typical value of $s = 30$ seconds and $1/\lambda_t = 15$ days, p is so close to zero as to be negligible.

4 MESSAGE LATENCY

The group communication protocol provides latency guarantees as well as reliability. The TSAE protocol only guarantees eventual delivery, but in practice messages propagate to every principal rapidly.

If information is propagated quickly, clients using different principals will not often observe different information, and loss of an update from site failure will be unlikely.

Simulation method

We constructed a discrete event simulation model of the TSAE protocol to measure propagation latency. The latency simulator measured the time required for an update message, entered at time zero to propagate to all available principals. The time required to send a message from one principal to another was assumed to be negligible compared to the time between anti-entropy sessions. The simulator could be parameterized to use different partner selection policies and numbers of sites. The simulator was run until either the 95% confidence intervals were less than 5%, or 10 000 updates had been processed. In practice 95% confidence intervals were generally between 1 and 2%.

The simulation modeled only the TSAE protocol, and did not consider the effect of combining TSAE with a best-effort multicast. Therefore the results in this section represent worst-cast behavior that would be improved if a multicast were added.

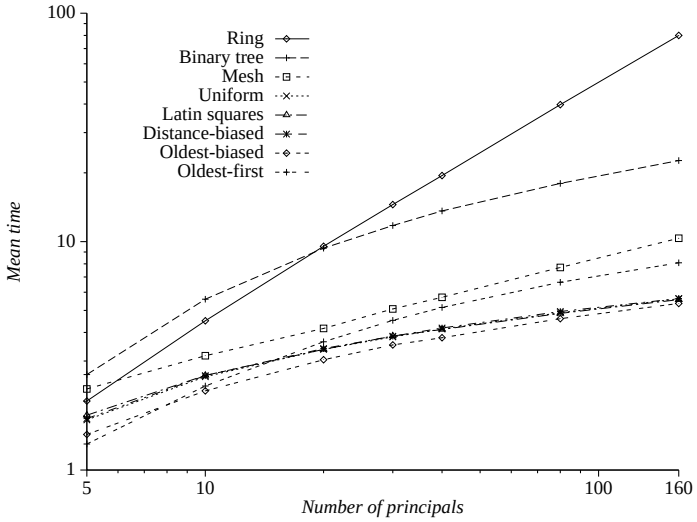

Figure 3 Effect of partner selection policy on scaling of propagation time.

Results

The partner selection policy also affects the speed of message propagation. Figure 3 shows the mean time required to propagate a message to every principal for several policies as the number of sites increases. The **uniform, latin squares,** and **distance-biased** policies give essentially identical performance. **Age-biased** appears to provide slightly better performance, which would appear to contradict the claim by Alon et al. that the **latin squares** policy is fastest [2]. We believe the difference arises from a slight difference in implementation: Alon's implementation requires that every principal propagate messages in well-defined rounds, while this simulation allows propagation to occur at random intervals. This may mitigate some of the benefit derived from Alon's **latin squares** policy. The policies that simulate a fixed topology – **ring, mesh,** or **binary tree** – have the worst performance and scaling.

These results indicate that simple random policies, such as uniform selection or age biasing, perform quite well.

5 TRAFFIC

A group communication system must not overload the network on which it

operates. This is particularly important if the group is to scale to a large size. The *traffic* induced by a system, as measured by the number of network packets that are sent and by the distance the packets traverse, is the primary measure of how the system will affect the network.

Researchers at Xerox PARC found that the original version of the Clearinghouse system overloaded their internetwork [4]. The original implementation combined anti-entropy sessions with a best-effort multicast, and used a uniform partner selection policy during anti-entropy. A revised implementation reduced the network load using a distance-biased partner selection policy.

Network traffic is measured by the number of packets sent, and by how many network links they must traverse. The number of packets is determined by the number of messages each principal sends, their size, and how often principals perform anti-entropy. In the absence of principal or network failures, the TSAE protocol ensures that every message is sent exactly once to each principal. The number of links each packet must cross is determined by the topology of the network and by the partner selection policy that principals use when performing anti-entropy.

For this performance evaluation, we introduced two partner selection policies in addition to those discussed in Section 2. The **cost-biased** policies preferentially select low cost partners. This is different from **distance-biased** selection when network links have different costs. The **cost-biased** policy selects a principal with probability proportional to the inverse of the difference between its cost and the highest cost in the group. The **cost-squared-biased** selects with probability proportional to the difference squared, increasing the probability that low-cost partners will be selected. The advantage of these policies is that they are not based on an arbitrary topology, but rather upon observable performance measures. This makes them appropriate for use in a wide-area internetwork where topological information is likely to be unavailable.

Simulation method

The system was modeled in a discrete-event simulation, written using a locally-developed library of C++ classes. Principals initiated anti-entropy sessions according to a Poisson process. Each run of the simulation performed 1000 anti-entropy sessions, collecting link traffic and cost statistics. At least 100 runs were performed, and they were repeated either until 1000 runs completed or until a batch-means analysis indicated that the 95% confidence interval width for each measure was less than 1% of the measured value. A complete set of

runs required between half an hour and six hours on a DECstation 5000/200, depending on the partner selection policy.

The simulator programs modeled a particular physical network topology. Principals were grouped into 5-cliques, and the cliques were connected by gateways to a backbone ring. This topology was selected not because it specifically modeled part of the Internet, but because it is representative of topologies that can concentrate traffic onto a few links. It is similar to the structure of the Internet today: regions of high connectivity, with regions weakly connected through a backbone network. Communication within a region is fast, while communication between regions can be expensive. Demers et al. noted that this kind of topology was a problem for the original Clearinghouse protocols [4].

Each simulator run was parameterized by the partner selection policy. The binary tree and mesh partner selection policies were not tested because they were patently inappropriate on ring-like topologies. The number of principals was fixed at 30 (six 5-cliques) but the cost of the backbone links relative to the intra-clique cost could be specified. This cost ranged from 1 through 160. The upper bound was chosen arbitrarily; it represents more than an order of magnitude difference from the lowest cost.

Results

Figure 4 shows the average cost of the links traversed by an anti-entropy session. As on a simple ring, the **uniform, oldest-biased, oldest-first,** and **latin squares** policies all performed about the same, while distance-biasing improves performance somewhat. The **ring** policy is somewhat better than all of these, though it scales in about the same way. The cost-biasing protocols produce somewhat more traffic when the backbone cost is low, but scale better as the backbone cost increases.

The cost-biasing policies decrease the traffic per link as the cost of the backbone increases. Figure 5 shows that the cost decrease is achieved by concentrating communication within a clique. When backbone links cost 80 times as much as a clique link, the **cost-biased** policy only allows between 1 and 2% of all anti-entropy sessions to cross between cliques. Fewer than 0.3% of all sessions cross backbone links when cost-squared-biasing is used. Clearly cost-biasing ensures that communication is predominantly local.

While one might want to reduce network traffic as much as possible, a reduction in traffic generally requires an increase in the time required to propagate a

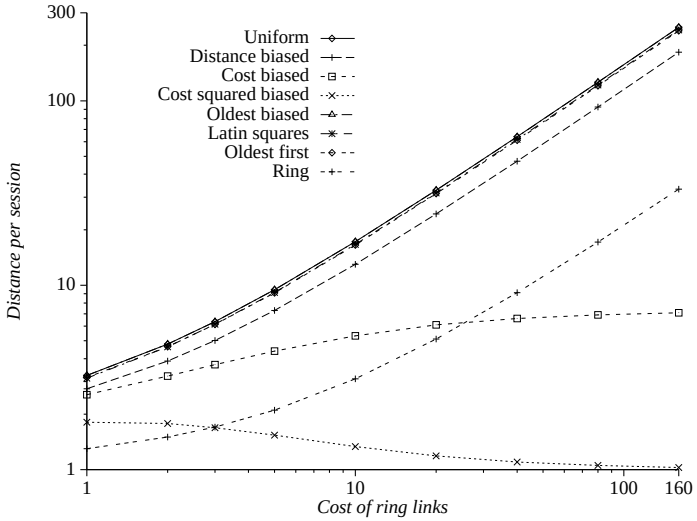

Figure 4 Effect of partner selection policy on the average number of network links used in an anti-entropy session. Measured on a ring of six 5-cliques. The cost of the backbone ring links varied from 1 to 160.

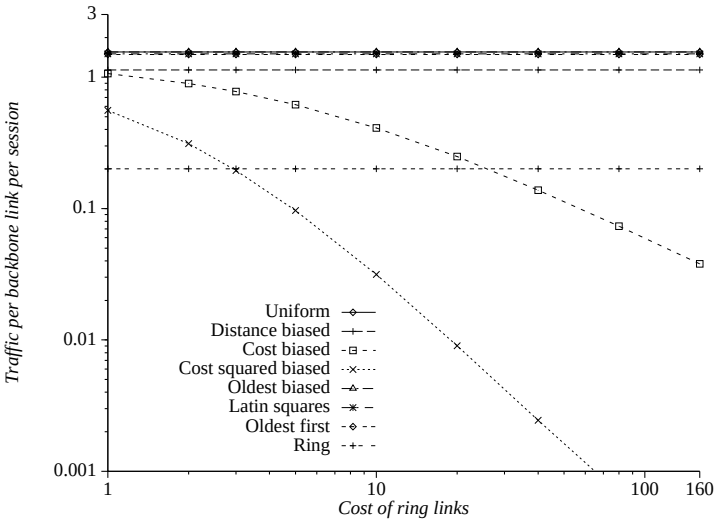

Figure 5 Effect of partner selection policy on the mean traffic per backbone ring link. The cost of the backbone links varied from 1 to 160. Only the cost-biased policies adapt to the cost of backbone links.

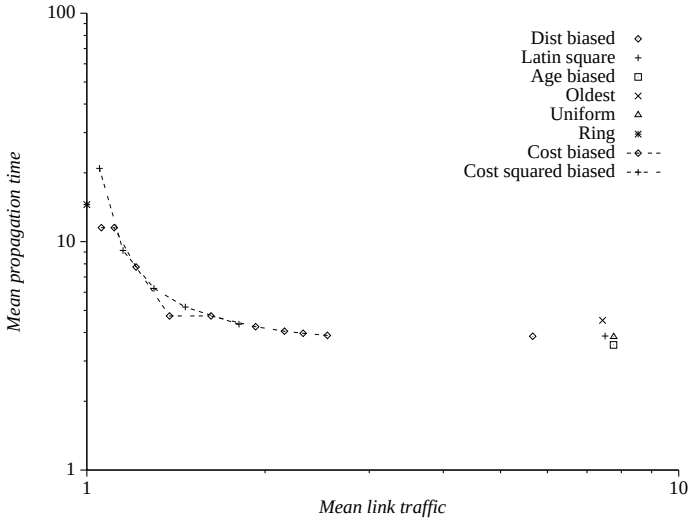

Figure 6 Scatter plot of the relationship between link traffic and propagation delay. Measured for 30 principals. Time is measured in multiples of mean time between anti-entropy sessions.

message. Figure 6 shows the relationship between the two. The time required to propagate a message is plotted as a function of the link traffic for each of several partner selection protocols. Every policy except **cost-biased** and **cost-squared-biased** is represented by a single point. Multiple points are reported for the cost-biased policies, showing how they respond to different backbone link costs. All of the policies excepting the **ring** policy appear to fit on a smooth curve.

6 CONSISTENCY

Weak consistency protocols allow principals to contain out-of-date information. There are two related measures of this effect – one concerning the propagation latency of a single message, the other concerning the consistency of group state. The likelihood that a principal is out-of-date with respect to other principals, and the difference between them, aggregates the effects of several messages.

Simulation method

A discrete event simulation modeled the TSAE protocol to measure information

age. The simulator used five events: one each to start and stop the simulation, one to send a message, one to perform anti-entropy, and one to sample the state of a principal. The simulation was first allowed to run for 1 000 time units so it would reach steady state, then measurements began. The simulation ended at 50 000 time units. Read, write, and anti-entropy events were modeled as Poisson processes with parameterizable rates. These rates were measured per principal. The simulator included different partner selection protocols and an optional unreliable multicast on writes.

The simulator maintained two data structures for each principal: the anti-entropy summary vector and a message number. It also maintained a global message counter. When a message was sent, the global counter was incremented and the sender's message number was assigned that value. If an unreliable multicast was being used, the message number would be copied to other principals if a the datagram was received. Anti-entropy events propagated message numbers between principals, as well as updating the principals' summary vectors.

Sampling events were used to collect measures of the expected age of data and the probability of finding old data. A principal was selected at random, and the message number for that principal was compared to the global counter. The difference showed how many messages the principal had yet to receive.

Our intent is to measure the consistency of *entire* databases. We measure the age of the state of a principal by the number of messages it has yet to receive. We expect most wide-area services to provide large databases, and so the messages will contain updates to many different database entries. The expected age, therefore, must be divided by number of data entries to arrive at the maximum probability that any particular entry is out-of-date.

The age of a principal's state depends on the ratio of the anti-entropy rate to the update rate for the state. The graphs in this section were generated using a mean time-to-update of 1 000 time units; the maximum anti-entropy rate investigated was only 200 times greater, giving a mean time-to-anti-entropy of five. This implies that all the results presented here are more pessimistic than would actually be observed.

Results

Our first analysis investigated the age of the information held by principals, and the degree to which it is affected by varying the rate of anti-entropy. Figure 7 shows the expected age of a member's state. Clearly, adding an unreliable

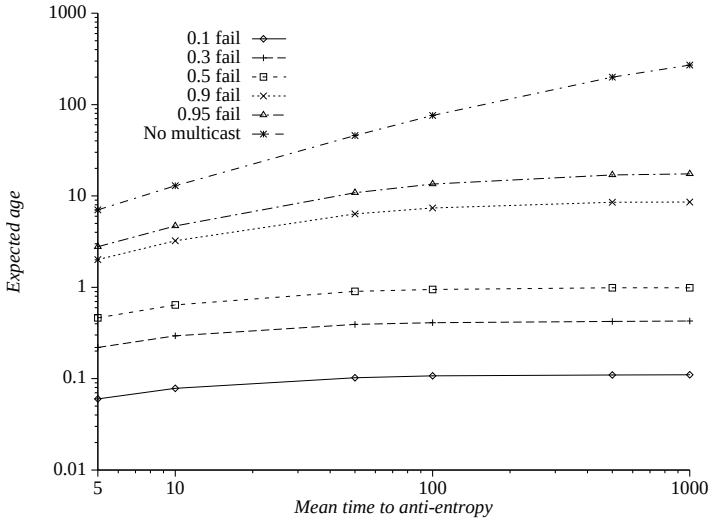

Figure 7 Expected data age as anti-entropy rate varies, for 500 principals. Mean time-to-write 1 000; **uniform** partner selection. Anti-entropy was combined with a best-effort multicast. The different curve show the effect of the probability of multicast message failure.

multicast on write significantly improves consistency. The message success probability is the most important influence on information age in large groups of principals. For small numbers of principals, increasing the anti-entropy rate dramatically improves both the probability of getting up-to-date information and the expected age.

Figure 8 shows how consistency depends on the number of principals. For these simulations the anti-entropy rate was fixed at 100 times that of writes. This value might be typical for a system where entries are corrected soon after being written. Later updates will be less frequent and the ratio will increase, improving the consistency. Once again an unreliable multicast provides considerable improvement.

We also investigated the effect of partner selection policy on information age. The results ranked partner selection policies in exactly the same order as they were ranked for propagation latency (Figure 3). The topological policies (**ring, binary tree, and mesh**) propagate more slowly, and give a greater expected age, than other policies. The other policies are nearly equal, though **oldest-first** has a slight advantage.

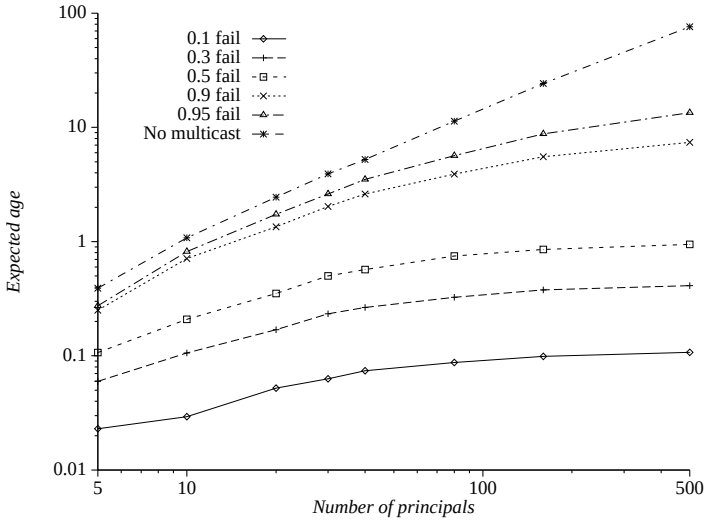

Figure 8 Expected data age as the number of principals varies, with anti-entropy occurring 100 times as often as writes. Uses **uniform** partner selection. Also shows the effect of varying message failure rates in a best-effort multicast.

7 CONCLUSIONS

We have evaluated the performance of the *timestamped anti-entropy* (TSAE) protocol, including network traffic, message latency, fault tolerance, and consistency. Simulation has proven a useful mechanism for investigating these goals. It would have been difficult to build and measure a system on the Internet with hundreds of replicas, particularly if we wanted to evaluate the protocols under specific failure conditions.

The analysis shows that the TSAE message delivery protocol scales well to large groups of principals. The time required to propagate an update from one principal to all others increases as the log of the size of the group, and the partner selection policy can be chosen to control network traffic as the membership grows.

Fault tolerance is the ability of a system to provide correct service even when some parts fail. The TSAE protocol provides excellent fault tolerance by delaying communication until a principal is available.

The negative aspect of weak consistency protocols is that principals will have out-of-date information. This investigation found that an unreliable multicast can mitigate most of this problem, and that at reasonable propagation rates principals are rarely more than a few updates behind.

The timestamped anti-entropy protocol meets our goals of scalability, reliability, and fault tolerance. It can scale to large groups without placing an undue load on the network, and the message delivery latency scales with the log of the group size. We find that the degree of inconsistency is very low, so that we believe this protocol to be ideal for many wide-area applications.

Acknowledgments

John Wilkes, of the Concurrent Systems Project at Hewlett-Packard Laboratories, and Kim Taylor, of UC Santa Cruz, assisted the initial development of these protocols. George Neville-Neil and Mary Long gave helpful comments on this work.

REFERENCES

[1] Agrawal, D., and A. Malpani, "Efficient dissemination of information in computer networks," Comp. Journal 34(6), Dec. 1991, pp. 534–41.

[2] Alon, N., A. Barak, and U. Manber, "On disseminating information reliably without broadcasting," Proc. 7th Int. Conf. on Distributed Computing Systems, 1987, pp. 74–81.

[3] Barbará, D., and H. Garcia-Molina, "The case for controlled inconsistency in replicated data," Proc. Workshop on the Management of Replicated Data, Nov. 1990, pp. 35–8.

[4] Demers, A., D. Greene, C. Hauser, W. Irish, J. Larson, S. Schenker, H. Sturgis, D. Swinehart, and D. Terry, "Epidemic algorithms for replicated database maintenance," Operating Systems Review 22(1), Jan. 1988, pp. 8–32.

[5] Emtage, A., and P. Deutsch, "Archie – an electronic directory service for the Internet," Proc. Winter 1992 Usenix Conf, pp. 93–110.

[6] Golding, R., "End-to-end performance prediction for the Internet," Tech. rep. UCSC–CRL–92–26, Computer and Information Sciences Board, University of California at Santa Cruz, Jun. 1992.

[7] Golding, R., "Weak-consistency group communication and membership," Ph.D. dissertation, published as Tech. Rep. UCSC–CRL–92–52, Computer and Information Sciences Board, University of California at Santa Cruz, Dec. 1992.

[8] Golding, R., and D. Long, "Design choices for weak-consistency group communication," Tech. rep. UCSC–CRL–92–45, Computer and Information Sciences Board, University of California at Santa Cruz, Sep. 1992.

[9] Golding, R., and K. Taylor, "Group membership in the epidemic style," Tech. rep. UCSC–CRL–92–13, Computer and Information Sciences Board, University of California at Santa Cruz, Apr. 1992.

[10] Ladin, R., B. Liskov, and L. Shrira, "Lazy replication: exploiting the semantics of distributed services," Operating Systems Review 25(1), Jan. 1991, pp. 49–55.

[11] Lamport, L., "Time, clocks, and the ordering of events in a distributed system," Comm. ACM 21(7), 1978, pp. 558–65.

[12] Long, D., J. Carroll, and C. Park, "A study of the reliability of Internet sites," Proc. 10th IEEE Symp. on Rel. Distr. Sys., Sep. 1991, pp. 177–86.

[13] Pu, C., and A. Leff, "Replica control in distributed systems: an asynchronous approach," Tech. rep. CUCS–053–090, Department of Computer Science, Columbia University, Jan. 1991.