



# Experimental Analysis of a Gossip-Based Service for Scalable, Distributed Failure Detection and Consensus

KRISHNAKANTH SISTLA, ALAN D. GEORGE\* and ROBERT W. TODD

*High-performance Computing and Simulation (HCS) Research Laboratory, Department of Electrical and Computer Engineering, University of Florida, P.O. Box 116200, Gainesville, FL 32611-6200, USA*

**Abstract.** Gossip protocols and services provide a means by which failures can be detected in large, distributed systems in an asynchronous manner without the limits associated with reliable multicasting for group communications. Extending the gossip protocol such that a system reaches consensus on detected faults can be performed via a flat structure, or it can be hierarchically distributed across cooperating layers of nodes. In this paper, the performance of gossip services employing flat and hierarchical schemes is analyzed on an experimental testbed in terms of consensus time, resource utilization and scalability. Performance associated with a hierarchically arranged gossip scheme is analyzed with varying group sizes and is shown to scale well. Resource utilization of the gossip-style failure detection and consensus service is measured in terms of network bandwidth utilization and CPU utilization. Analytical models are developed for resource utilization and performance projections are made for large system sizes.

**Keywords:** cluster computing, consensus, failure detection, fault tolerance, gossip protocol, layering

## 1. Introduction

With the constantly increasing performance levels and decreasing costs associated with uniprocessor and multiprocessor servers and desktop computers, high-performance cluster systems such as *Cplant* at Sandia National Labs hold the potential to provide a formidable supercomputing platform for a variety of grand-challenge applications. However, in order for heterogeneous and high-performance clusters to achieve their potential with parallel and distributed applications, a number of technical challenges must be overcome in terms of scalable, fault-tolerant computing. These challenges are made all the more complex with the increasing heterogeneity of technologies in network architectures, server architectures, operating systems, sharing strategies, storage subsystems, etc.

One particular need is the capability for large-scale, heterogeneous clusters based on open systems strategies to achieve failure detection and consensus in a scalable fashion. Applications capable of self-healing, perhaps with checkpointing, process migration, etc., require such services so that jobs involving many resources in a large-scale system can come to a consensus on the state of the system as nodes dynamically exit and enter operational status in the system. Classical group communications are inappropriate due to their inherent limits in scalability, and proprietary vendor solutions do not support the heterogeneous nature of the system.

A particularly promising approach towards this goal leverages the notion of gossip communication. Random gossiping, which is based on the individual exchange of liveness information between nodes, has been investigated as a pos-

sible failure-detection approach for scalable heterogeneous systems [1–3]. This interest is due to several advantages of gossip-style failure detectors. For example, gossiping is resilient and does not critically depend upon any single node or message. Moreover, such protocols make minimal assumptions about the characteristics of the networks and hosts, and hold the potential to scale with system size.

In this paper, experimental results from the implementation of a new failure detection and consensus service based on gossiping are presented and analyzed. Results with several forms of gossiping are included and compared, based on random and round-robin intercommunication patterns, and a single-level or *flat* scheme is compared with a hierarchical or *layered* scheme. Resource utilization of the failure-detection and consensus service is measured in terms of network utilization and CPU utilization. Analytical models for resource utilization are developed and projections are made for large system sizes.

The remainder of this paper is organized as follows. In section 2, a brief background is provided on concepts in gossip-based failure detection and consensus. Section 3 describes the flat and layered gossip schemes. Experimental results for different gossip schemes are presented in section 4, while the section 5 details the analytical models for resource utilization and the performance projections for large system sizes. Finally, section 6 offers conclusions to the work presented and directions for future work.

## 2. Background

Van Renesse et al. [1] first investigated gossiping for failure detection. In their paper, they present three protocols: a basic version, a hierarchical version, and a protocol that

\* Corresponding author.  
E-mail: george@hcs.ufl.edu

Table 1  
Characteristics of deterministic gossip protocols.

Protocol	Destination ID	Minimum $T_{\text{cleanup}}$
Round-robin	Destination ID = source ID + $r$ , $1 \leq r \leq n$	$T_{\text{cleanup}} \geq \alpha T_{\text{gossip}}$ , where $\frac{\alpha(\alpha-1)}{2} + 1 \leq n \leq \frac{\alpha(\alpha+1)}{2} + 1$
Binary round-robin	Destination ID = source ID + $2^{r-1}$ , $1 \leq r \leq \log_2(n)$	$T_{\text{cleanup}} \geq (\log_2 n) T_{\text{gossip}}$

deals with arbitrary host failures and partitions. Burns et al. conducted a study that focused on a simulative performance analysis of several gossip protocols operating on a distributed systems model comprised of clusters of nodes connected by Myrinet [2]. Although the basic protocol is sufficient in a small system, it was found to yield poor scalability with increase in system size.

The hierarchical protocol makes use of *a priori* knowledge of the underlying network topology to send a majority of gossips locally. A piggyback protocol was introduced that further reduces the network bandwidth required for gossiping by monitoring traffic generation patterns and piggybacking gossips on application-generated messages whenever possible. In their simulative study of the three protocols, Burns et al. showed that the hierarchical and piggyback protocols were found to be significantly more scalable than the basic protocol for gossiping.

The inefficiency of the basic protocol can be addressed by introducing patterned communication among the gossiping members. In [3], Ranganathan et al. introduced several efficient communication patterns and simulated their performance. The round-robin protocol improves on basic randomized gossiping by distributing gossip messages in a deterministic order that optimizes bandwidth consumption. Redundant gossiping is completely eliminated in the binary round-robin protocol. This section briefly describes the gossip protocols [1–3] and the distributed consensus algorithm [3], which form the basis for this work.

Three of the key parameters involved in the failure detection and consensus service are the gossip time, the cleanup time, and the consensus time. Gossip time, or  $T_{\text{gossip}}$ , is the time interval between two consecutive gossip messages. Cleanup time, or  $T_{\text{cleanup}}$ , is the time interval after which a node is suspected to have failed. Finally, consensus time, or  $T_{\text{consensus}}$ , is the time interval after which consensus is reached about a failed node. The first two are input parameters configured for a particular gossip-based failure detection system. The cleanup time is some multiple of the gossip time, where the time required for information to reach all other nodes sets the lower bound for  $T_{\text{cleanup}}$  (referred to herein as *minimum*  $T_{\text{cleanup}}$ ). When both values are relatively small, the system is more quickly responsive to changes in node liveness. When they are relatively large, response is slower but resource utilization decreases. The third parameter is a performance metric determining how quickly failures are detected and consensus is reached.

## 2.1. Random and deterministic gossip protocols

Gossip is a scalable means of disseminating information in a distributed system. Due to its distributed nature, gossip is resilient to multiple failures in the system. Gossiping can be broadly classified into two categories: *random* and *deterministic*. In random gossiping, each node gossips to a randomly selected destination every  $T_{\text{gossip}}$  seconds. The random form of gossiping, also known as basic gossip, was first studied in [1]. While having the benefit of simplicity, the nature of random gossip implies that redundant communication will occur whereby state information is received that has already been received before. To address this limitation and others, deterministic algorithms define a predetermined communication pattern among the gossiping nodes. More efficient forms of gossip, namely round-robin (RR) and binary round-robin (BRR), were first studied in [3]. These two protocols are summarized and compared in table 1 in terms of minimum  $T_{\text{cleanup}}$  value, the relationship between source and destination nodes, the round ( $r$ ), and the number of nodes in the system ( $n$ ).

### 2.1.1. Basic gossip

With the basic or random gossip, every  $T_{\text{gossip}}$  seconds each node randomly selects another node and transmits a heartbeat list and suspect matrix. There is no upper bound on the time it takes for a quantum of information to reach all other nodes. In addition, redundant communication typically occurs, hence wasting bandwidth. Conversely, the basic protocol is resilient to network and host failures and is easy to implement.

### 2.1.2. Round-robin (RR) gossip

In the basic gossip protocol it is possible for a node not to receive a gossip for such a long enough period of time that false failure detections can occur. The RR protocol is a deterministic gossip protocol aimed at reducing both redundant communication and false failure detections. In the RR gossip protocol, gossiping takes place in definite rounds every  $T_{\text{gossip}}$  seconds. Every round, each node will receive and send a single gossip message. Using this protocol, the number of rounds needed to establish one-to-one communication with every other node in the system is  $n - 1$ .

The RR gossip protocol guarantees that all nodes will receive an arbitrary node's updated heartbeat value in a bounded time. This trait allows us to set a lower limit to the value of  $T_{\text{cleanup}}$ . While the RR protocol reduces  $T_{\text{cleanup}}$  by adopting a deterministic approach, it eliminates some but not all redundant communication.

### 2.1.3. Binary round-robin (BRR) gossip

The communication redundancy inherent in the RR protocol is eliminated in the BRR gossiping protocol. If the number of nodes in a system is a power of two, by doubling the number of nodes that receive the state information every  $T_{\text{gossip}}$ , state information can be disseminated throughout the system with no redundant communication. This observation is leveraged in BRR, however the simplest form of the BRR protocol works only for a system size that is a power of two and extensions are required for incremental changes in the system size.

## 2.2. The consensus algorithm

Each node maintains three data structures: a gossip list, a suspect vector and a suspect matrix. These three structures together represent the perceived state of each node within a system. The gossip list is a vector of size  $n$ , where  $n$  is the system size. Element  $j$  of the gossip list on node  $i$  contains the number of  $T_{\text{gossip}}$  intervals, termed the heartbeat value, since node  $i$  has received a gossip from node  $j$ . If the value in element  $j$  is greater than  $T_{\text{cleanup}}$ , then the corresponding element of the suspect vector is set to '1', otherwise it remains '0' indicating a healthy node.

The suspect vectors of all  $n$  nodes are joined together to form a suspect matrix of size  $n \times n$ . From each node a gossip is sent every  $T_{\text{gossip}}$  seconds containing the suspect matrix. On receipt of a gossip message, the state view of a node is updated by merging the data structures as explained in [3]. Consensus is reached on the state of node  $j$  if each element in column  $j$  of the suspect matrix representing unsuspected nodes contains a '1'. When a node detects that consensus has been reached, it sends a broadcast message to all nodes in the system informing them of the event.

## 3. Layering and consensus propagation

The gossip protocol presented in its original form can be referred to as flat gossiping since all the nodes in the system are treated as a single group or layer. An alternative to flat gossiping is layered gossiping, where nodes in the system are divided into groups, gossiping takes place within groups, and one or more members of each group take part in gossiping between groups in a hierarchical framework. Tradeoffs are expected between flat and layered gossiping in terms of simplicity and scalability of consensus time and resource utilization, and these tradeoffs are explored in the experiments to come.

The mechanism by which consensus information is distributed in the system plays a crucial role in determining the scalability of consensus time. On detecting that consensus is reached, a node will either broadcast this information to the rest of the system, or it will use gossip messages to propagate consensus information. The former approach is termed as consensus with broadcast and the latter, consensus without broadcast. A combination of these gossiping alternatives results in several possible gossip schemes. This section briefly

describes the gossiping alternatives and the resulting combinations.

The distributed consensus algorithm works efficiently for small systems. However, the size of the suspect matrix and the gossip list dramatically grows with the system size leading to both implementation problems and an increase in communication and processing overhead. Additionally, due to the increase in the minimum value of  $T_{\text{cleanup}}$ , failure detection time also increases dramatically. These problems can be addressed by layering groups of clusters into a hierarchical system.

Using the layered gossiping protocol, the total number of nodes is divided into groups or clusters in a hierarchy of two or more levels or layers, typically chosen to fit the network topology. In this setup, each node gossips only within its local group. One or more higher layers handle gossiping between the groups. In the case of a two-layer system, nodes in the lower-layer (L1) take turns to participate in the upper-layer (L2) gossip. When consensus is reached within a lower-layer group, this information is propagated to all nodes in all groups. An example of a two-layer approach is shown in figure 1. Layering groups of systems in this manner also facilitates increased heterogeneity in the system by grouping similar nodes together. The separate layers might have different gossiping parameters, as network partition failures are not as frequent as individual failures. Layering also improves the scalability of bandwidth and CPU utilization, since the suspect matrix now grows more slowly with system size.

Consensus information can be propagated using either broadcast or gossip messages. A special consensus message containing the *id* of the faulty-node is broadcast to the whole system, if consensus with broadcast is used. In systems that support a true hardware broadcast, all nodes in the system reach consensus around the same time with little variation in consensus time. If consensus is propagated by gossip messages, the *id* of the faulty-node is attached to the gossip mes-

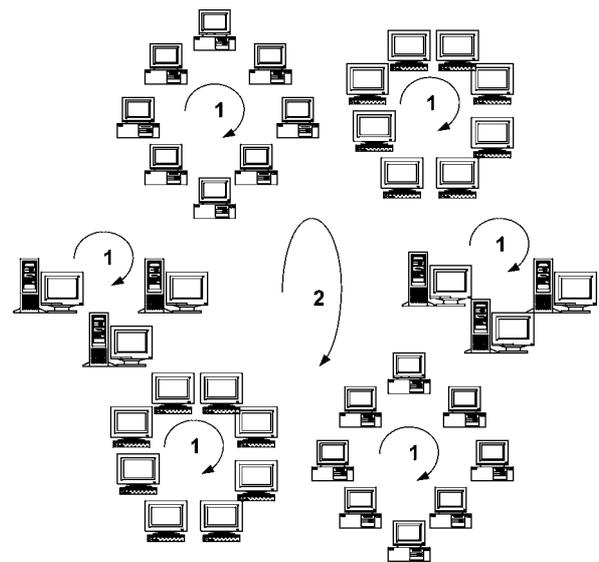


Figure 1. Layered implementation of gossiping in a heterogeneous system: 1 – the lower layer, for detecting node failures and 2 – the upper layer, for detecting network partitions.

sages. Higher-layer gossip messages carry this information to other groups in a layered system. There is appreciable variation in consensus time across different nodes.

Combined with the two possible schemes of flat and layered gossip, we have four possible combinations namely *flat with broadcast* (FWB), *flat without broadcast* (FWOB), *layered with broadcast* (LWB) and *layered without broadcast* (LWOB). In this paper we study these four combinations with respect to consensus time, resource utilization and scalability.

#### 4. Experiments and results

Experiments were conducted using a heterogeneous PC cluster comprised of over 100 nodes that features a control network of switched Fast Ethernet and a variety of high-speed data networks including SCI, Myrinet, cLAN, and Gigabit Ethernet. The gossip-style failure detection and consensus service was implemented as a standalone daemon on each of the nodes. Individual nodes communicate over the control network using the UDP transport. Each node is installed with the Redhat Linux V6.1 or V6.2 operating system and a version 2.2 kernel. In this section we present the experimental results from consensus time, network utilization and CPU utilization experiments for the gossip schemes previously enumerated.

##### 4.1. Consensus time experiments

At the beginning of each experiment, one node is selected randomly as the faulty node. The faulty node stops gossiping at a fixed time between 0 and  $1000T_{\text{gossip}}$  seconds. The remaining nodes reach consensus on the faulty node and the consensus time is recorded. Each of the results provided represent the average of five repeated cases, each choosing a different node to stop responding at a different time. The variance between repeated experiments was observed to be insignificant. A value of 10 ms for  $T_{\text{gossip}}$  is used for all experiments.

In the first set of consensus experiments we study the effect of different parameters on flat gossiping and compare the performance of FWB and FWOB. The effect of  $T_{\text{cleanup}}$  on the consensus time for a 16-node system employing FWB is shown in figure 2(a). It is noted that consensus time decreases as  $T_{\text{cleanup}}$  decreases, until a minimum cleanup time below which true consensus cannot be reached. If a value for  $T_{\text{cleanup}}$  were to be selected below this minimum, false failure detections will increase and make consensus impossible. For values of cleanup time higher than the minimum, consensus time scales linearly with all three protocols and they share common performance characteristics. The minimum cleanup time is proportional to the number of rounds required to disseminate gossip messages to all of the members in the system. The BRR protocol requires the least number of rounds to disseminate gossip messages and hence provides the lowest minimum  $T_{\text{cleanup}}$ . By contrast, the basic protocol requires the largest number of rounds and hence has the highest minimum  $T_{\text{cleanup}}$ .

Figure 2(b) shows the variation of minimum  $T_{\text{cleanup}}$  when the system size is varied from 8 to 96 nodes. With each of the protocols, it is observed that the minimum cleanup time versus system size increases at a modest rate in the region of interest. An interesting observation is the crossover between the basic and RR protocols at a system size of 64. This behavior is attributed to a lack of clock synchronization between nodes. Since no attempt was made to synchronize the participating nodes, the round-robin communication pattern deviates from the ideal with an increase in the number of nodes.

In figure 3, the best consensus times for different system sizes are presented for FWB and FWOB systems. In doing so, the lowest possible consensus time is achieved in each case by setting the  $T_{\text{cleanup}}$  parameter to its minimum value for that system size. The results indicate that the basic protocol outperforms RR for system sizes larger than 72 for the same reasons as explained above. On average, all three protocols exhibit a linear scalability in consensus time versus system size in the region of interest. It is observed that for the

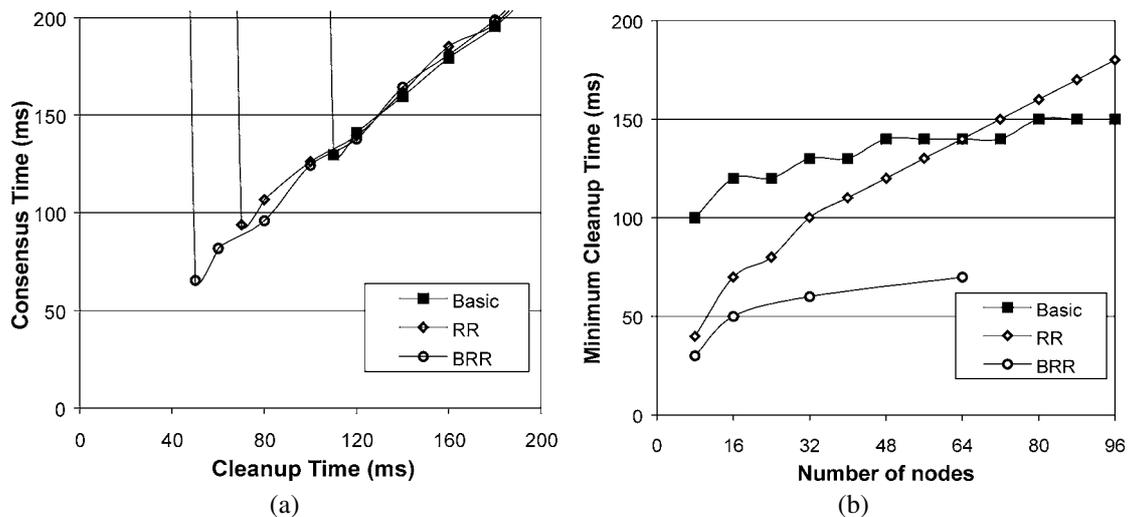


Figure 2. Impact of cleanup time on consensus in a 16-node FWB system (a), and the relationship between minimum cleanup time and system size in a FWB system (b).

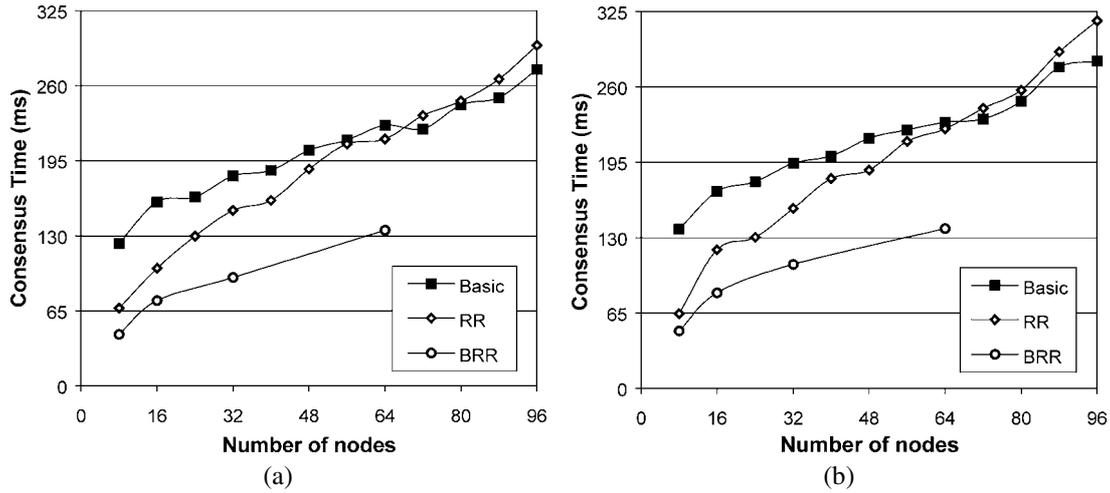


Figure 3. Scalability of consensus time in a flat system: (a) FWB and (b) FWOB.

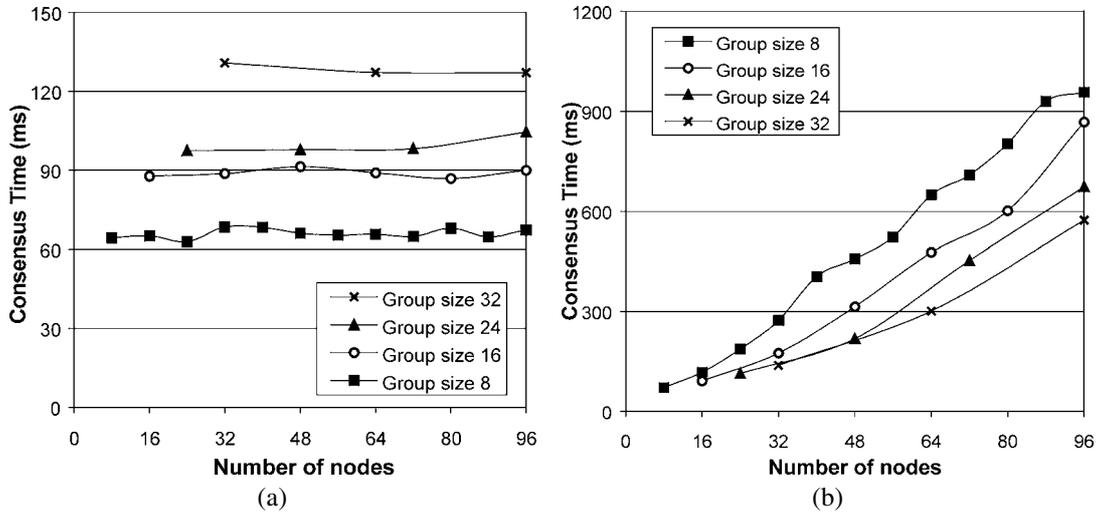


Figure 4. Scalability of consensus time in a layered system for several group sizes (where L1 is RR and L2 is basic): (a) LWB and (b) LWOB.

same system size and cleanup time, FWB has a marginally lower consensus time than FWOB. In FWB, the node that first detects consensus broadcasts the information to all other nodes, thus the consensus time is approximately the time for first detecting consensus. Conversely, in FWOB all nodes detect consensus through gossip messages, which arrive after the first detection of consensus, thus FWOB exhibits a marginally higher consensus time than FWB.

In the second set of consensus experiments, we divide the system into a two-layer hierarchy with the lower layer (L1) employing the RR gossiping protocol and the upper layer (L2) using the basic protocol. All of the consensus times presented are obtained by setting  $T_{\text{cleanup}}$  to the lowest possible value for that system size unless otherwise noted.

In the first experiment in this set, the number of nodes in the layered system is varied and consensus time is measured using several different group sizes. Figure 4 shows the results of this experiment for LWB and LWOB systems.

For LWB, shown in figure 4(a), consensus time is observed to be virtually independent of system size, achieving ideal scalability in the region of interest. While the slopes remain

approximately flat, the magnitudes shift upward with an increase in the group size, meaning that smaller groups achieve higher performance. This behavior is to be expected since the failure detection and consensus on a failed node within a group needs to be reached only within that group and hence is independent of system size, and the time for propagation of consensus is also relatively constant.

By contrast, in LWOB the consensus time increases with system size at a significant slope. Moreover, as shown in figure 4(b), in this case the larger groups achieve better performance than the smaller ones. This behavior is to be expected since, for a fixed system size, smaller-sized groups imply a larger number of groups to which consensus information needs to be propagated by gossip, thereby increasing the time to reach consensus.

In figure 5, the consensus time is measured for a varying number of groups while keeping the system size fixed at 96 nodes. For LWB, the previous experiment showed that the smaller group sizes should result in faster overall consensus time, which is the case here as well. However, when a fixed value of  $T_{\text{cleanup}}$  is used (i.e., 200 ms in this case), an in-

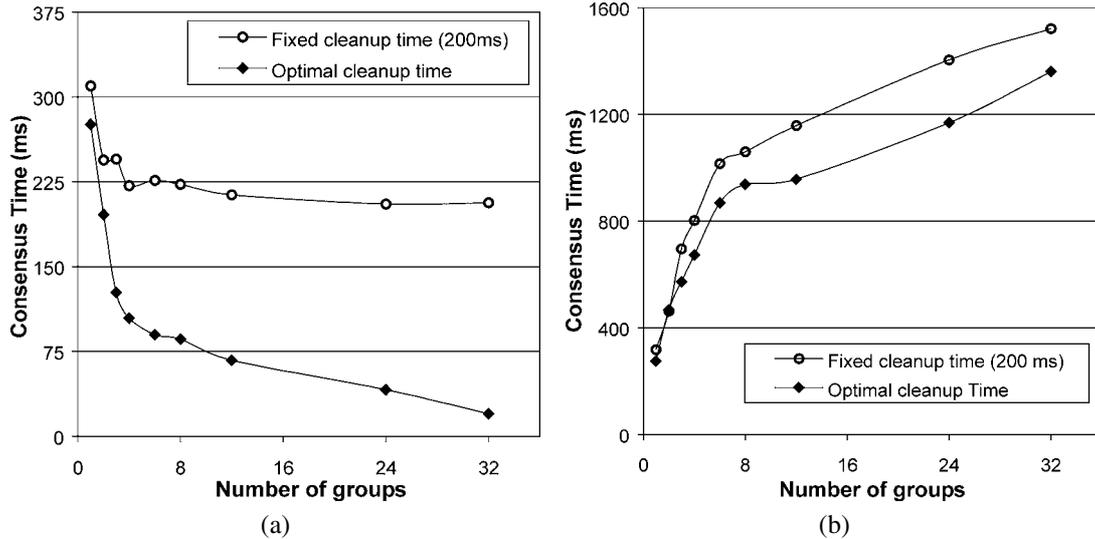


Figure 5. Impact of number of groups on consensus time in a layered system of 96 nodes: (a) LWB and (b) LWOB.

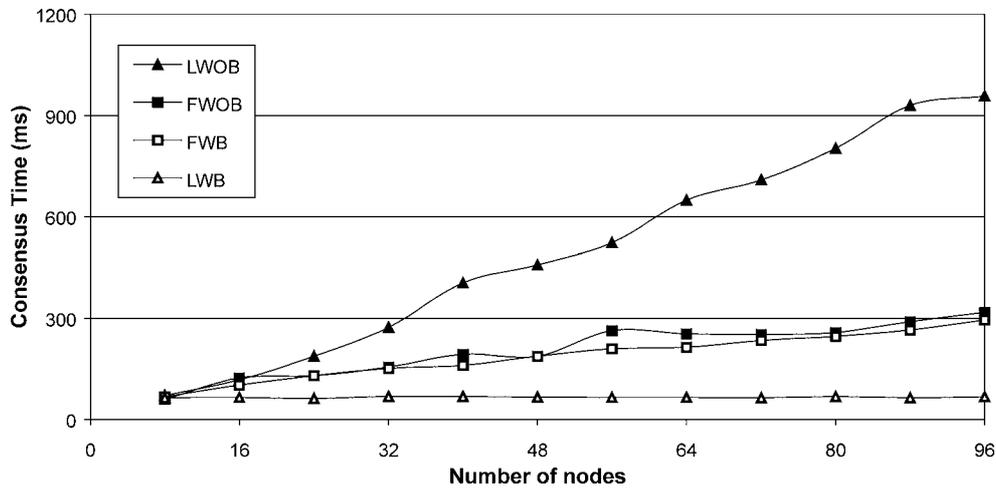


Figure 6. Comparison of scalability of consensus time in layered and flat systems.

crease in the number of groups exhibits a diminishing degree of improvement in consensus time. This behavior can be expected, since as the groups become smaller the optimal value of cleanup time does as well and thus a fixed cleanup time results in needless overhead. By contrast, when the  $T_{\text{cleanup}}$  value is adapted for each distinct group size, the consensus time decreases substantially with an increase in the number of groups.

Although these results make it clear that, in terms of consensus time for LWB, the optimal group size is the smallest (e.g., group size of three in this case to tolerate a single faulty node), other requirements may dictate otherwise. For example, as discussed in sections 4.2 and 4.3, network and CPU utilization considerations are found to favor a balance in size of groups versus number of groups. Moreover, the issue of reliability and tolerance of multiple faults within a group is also of key concern is selecting the group size. Since the consensus protocol functions with a simple majority vote, the use of smaller group sizes will imply tolerance of fewer faults per group (i.e., one fault can be tolerated in a three-node group,

two in a five-node group, etc.). The reliability issues associated with the selection of group size are left for future work.

As expected, LWOB behaves in a manner opposite to that of LWB. In LWOB, the increase in the time taken to spread consensus information to a larger number of groups dominates the decrease in the consensus time obtained by a smaller group size. The optimal cleanup time does achieve a benefit, but to a lesser degree than with LWB due to the magnitude of the consensus propagation time.

Finally, figure 6 presents a scalability comparison of all four schemes. In this case, flat gossiping uses RR and layered gossiping is set up as in the previous experiments. The group size for layered gossip is set to eight.

As previously observed, LWB achieves the best performance and scalability. For example, the consensus time for a 96-node LWB system is only 25% that of a comparable flat system. Perhaps surprisingly, LWOB is the least scalable of all the schemes, with the consensus time for a 96-node LWOB system being approximately three times that of a comparable flat system and rapidly diverging. However, this poor per-

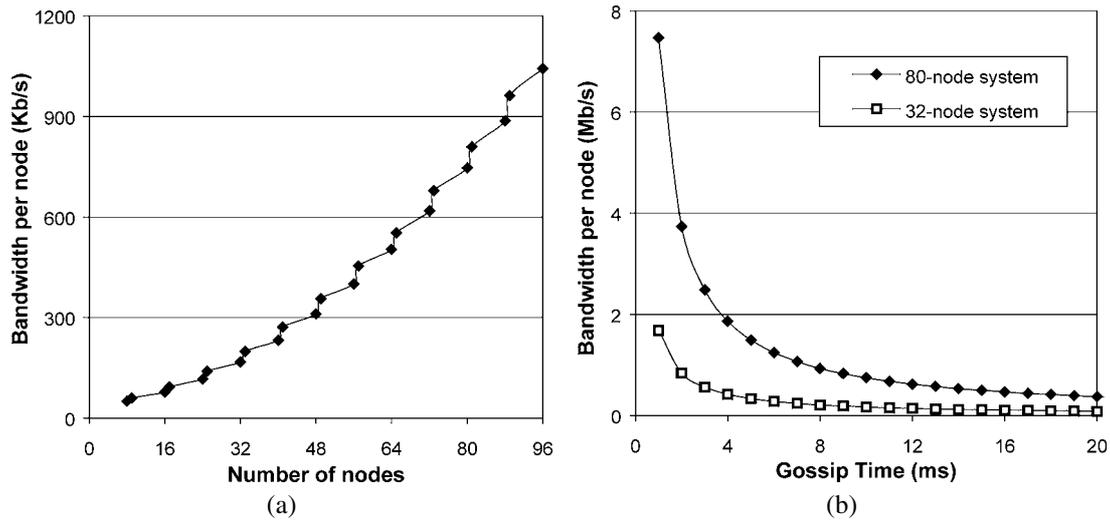


Figure 7. Scalability of network utilization per node in a flat system (a), and the effect of gossip time on network utilization in a flat system (b).

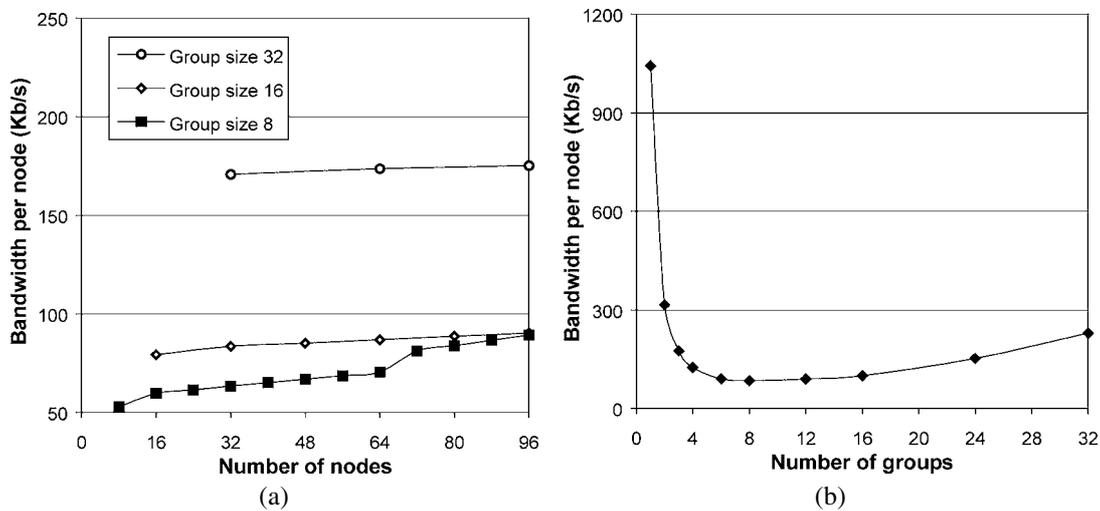


Figure 8. Scalability of bandwidth per node in a layered system for several group sizes (a), and the impact of number of groups on bandwidth per node in a layered system of 96 nodes (b).

formance is attributed to the limited scalability of L2 gossip with a fixed group size, which in turn increases the time required to propagate the consensus information throughout the system when a failure occurs. Of course, as the previous experiments show, a small number of large groups will reach consensus much faster in LWOB than will a large number of small groups.

#### 4.2. Network utilization experiments

Network bandwidth utilization of gossiping is measured on a per-node basis using *Ethereal*, a packet-capturing tool available in the public domain. *Ethereal* is used to capture all the gossip packets from a host over a fixed time period and estimate the network bandwidth consumed by them. Network utilization of FWB is virtually the same as that of FWOB, since consensus propagation is an infrequent phenomenon compared to gossiping. Similarly, LWB has virtually the same network utilization as LWOB for the same reason. A value of

10 ms for  $T_{\text{gossip}}$  is used for all experiments unless otherwise noted.

In figure 7(a), bandwidth per node is measured for a varying number of nodes in a flat system. It is observed that network bandwidth per node increases with an overall  $O(n^2)$  scalability, exceeding 1 Mb/s for a 96-node system. Poor scalability of flat gossiping is attributed to the increases needed in the size of the suspect matrix. Aggregate network bandwidth utilization of a flat system varies as  $O(n^3)$ , exceeding 100 Mb/s for a 96-node system. Figure 7(b) shows the variation of bandwidth per node in a flat system for different choices of  $T_{\text{gossip}}$ . It is observed that network utilization per node decreases hyperbolically with increase in the value of gossip time. With increases in gossip time, nodes gossip less frequently thus decreasing the network bandwidth needed.

For studying the network utilization of a layered system, nodes in the system are divided into two layers as described in the previous layered experiments. In figure 8(a), the variation of network utilization per node versus system size is shown

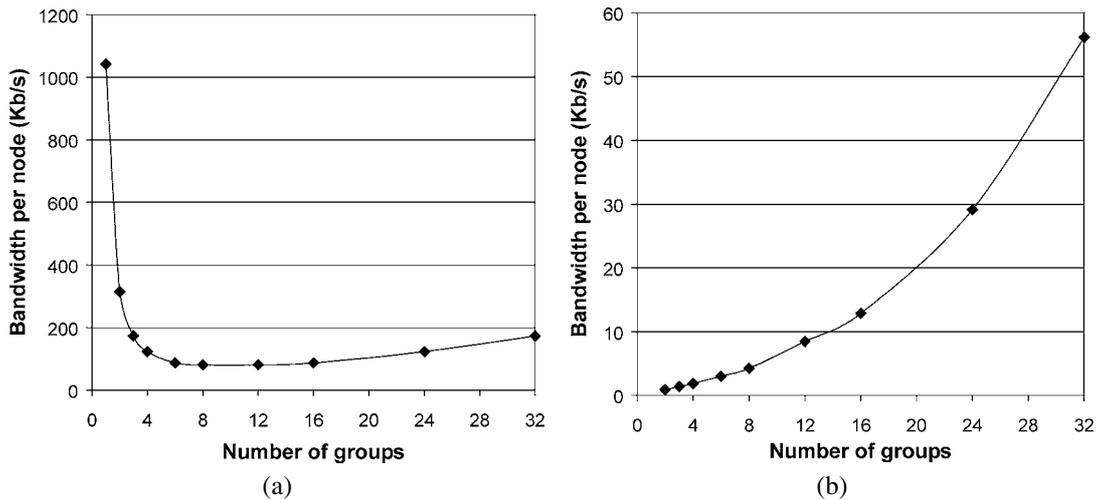


Figure 9. Impact of number of groups on network utilization in a layered system of 96 nodes: (a) L1 network utilization and (b) L2 network utilization.

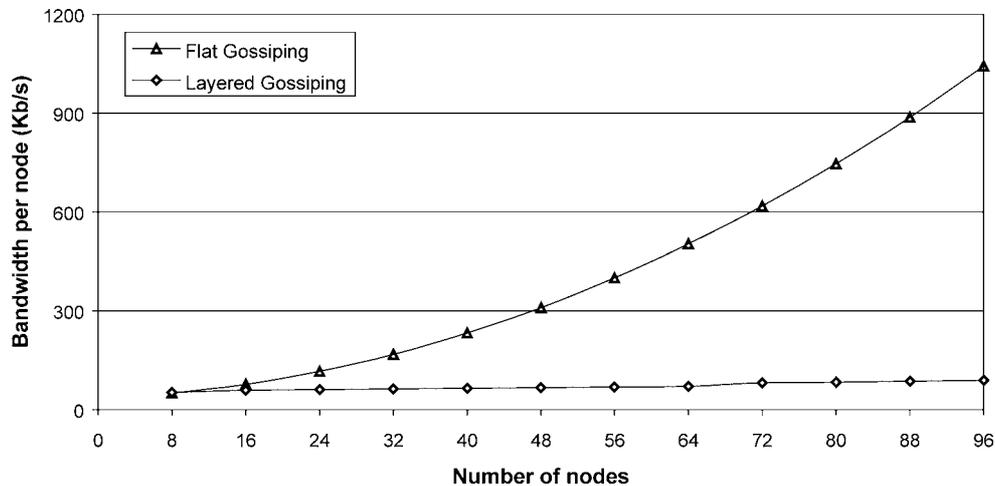


Figure 10. Comparison of scalability of network utilization in layered and flat systems.

for different group sizes. Network utilization per node in a layered system exhibits approximately linear scalability in the region of interest. It will be shown section 4.3 that the actual variation is more complex. For a fixed group size, an increase in system size leads to an increase in the number of groups, increasing the L2 component of network utilization while the L1 component remains constant. The improved scalability of layered gossip is due to the slower rate of increase of L2 network utilization with system size in the region of interest.

The crossover between the curves for different group sizes in figure 8(a) suggests that, for a given system size, there exists an optimum group size that minimizes network bandwidth per node. The existence of such an optimum group size can be confirmed by studying the variation of network utilization per node with the number of groups, while keeping the system size constant. Figure 8(b) shows the results of this experiment for a 96-node system. It is noted that network utilization per node initially decreases dramatically with increase in the number of groups, but increases when the number of groups exceeds 12. Thus, the group size that optimizes network utilization per node in a two-layered system of 96 nodes is 8.

Analytical models presented in the next section can be used to determine the optimum group size for any system size.

In figure 9, network bandwidth per node of L1 gossiping and L2 gossiping are shown separately. An L1 gossip packet contains both the first-layer and second-layer information. As the number of groups increases, second-layer information increases in size as  $O(g^2)$ , where  $g$  is the number of groups. Simultaneously, first-layer information in the L1 gossip packet is reduced due to the reduction in the number of nodes per group. Initially, the reduction in first-layer information dominates, thus reducing L1 network utilization. Eventually, increasing second-layer information dominates, increasing L1 network utilization after a certain number of groups. Conversely, L2 gossip packets contain only second-layer information, and thus L2 network utilization increases as  $O(g^2)$  as shown in figure 9(b).

Finally, figure 10 presents a scalability comparison of flat and layered gossiping. In this case, flat gossiping uses RR and layered gossiping is set up as in the previous experiments. The group size for layered gossip is set to eight. It is observed that layered gossip scales well compared to flat gossip, with

Table 2  
Summary of node architectures.

Node type	Configuration
Alpha	400 MHz Intel Celeron processor with integrated 128 KB L2 cache
Delta	600 MHz Intel Pentium-III processor with internal 512 KB L2 cache
Zeta	733 MHz Intel Pentium-III processor with integrated 256 KB L2 cache

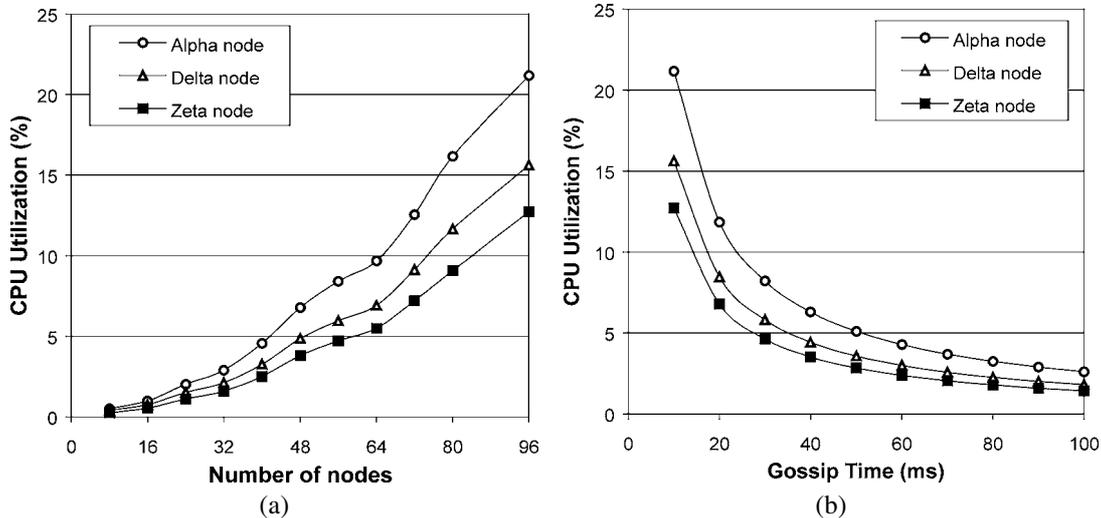


Figure 11. Scalability of CPU utilization in a flat system (a), and the effect of gossip time on CPU utilization in a flat system of 96 nodes (b).

the network utilization per node for a 96-node layered system being approximately 10% that of a comparable flat system. On average, the flat system exhibits an  $O(n^2)$  scalability, while bandwidth utilization in the layered system is virtually constant.

#### 4.3. CPU utilization experiments

CPU utilization of the failure detection service is computed by measuring the number of CPU cycles consumed per  $T_{\text{gossip}}$ . Experiments were conducted on three different node architectures, each with a different CPU configuration, and thus the results presented form a family of curves. The three node architectures used are summarized in table 2. For the same reasons as mentioned for network utilization experiments, FWB and FWOB exhibit the same processor utilization, as do LWB and LWOB. A value of 10 ms for  $T_{\text{gossip}}$  is used for all experiments unless otherwise noted.

Owing to the increase in the size of the suspect matrix and its processing in the detection and consensus algorithm, computational complexity of flat gossiping increases dramatically with increase in system size. Figure 11(a) shows the scalability of CPU utilization in a flat system. We observe that CPU utilization of a flat system exhibits approximately second-order scalability. It is estimated that processor utilization with flat gossiping will reach 100% for  $n \approx 250$ , thereby making flat gossiping with  $T_{\text{gossip}} = 10$  ms impractical at or even near this threshold.

In figure 11(b), the effect of the choice of  $T_{\text{gossip}}$  is shown for a 96-node system employing flat gossiping. For a value of  $T_{\text{gossip}}$  greater than 10 ms, CPU utilization decreases hy-

perbolically with increase in  $T_{\text{gossip}}$ . For  $T_{\text{gossip}}$  values less than 10 ms, processor utilization is approximately 100%, as the failure detection process needs to busy-wait on the CPU to achieve an accurate value of  $T_{\text{gossip}}$ .

A layered system is set up in the same manner as for the consensus time experiments. Figure 12(a) shows the scalability of CPU utilization in a layered system for several group sizes. Compared to flat gossiping, relatively insignificant processor usage is experienced with layered gossiping. In the region of interest, processor utilization of a layered system varies approximately linearly with system size, where the average slope of the linear variation reduces with an increase in group size. The difference in the slopes leads to crossovers between the curves for different group sizes. For example, for system sizes larger than 96, a group size of 16 has lower utilization than a group size of 8. The improved scalability of layered gossip is attributed to reduced L1 computational complexity and slow increase in L2 computation in the region of interest. In figure 12(b), CPU utilization of a layered system is shown for a varying number of groups while keeping the system size fixed at 96. The group size that optimizes processor utilization of a two-layered system of 96 nodes is observed to be 8.

Figure 13 shows the individual components of L1 and L2 processor utilization in a two-layered system. For a fixed system size, an increase in the number of groups causes reduction in L1 CPU usage due to a decrease in the number of nodes per group. Conversely, L2 CPU usage increases as a second-order polynomial with increases in the number of groups. In general with an increase in the number of groups, second-layer

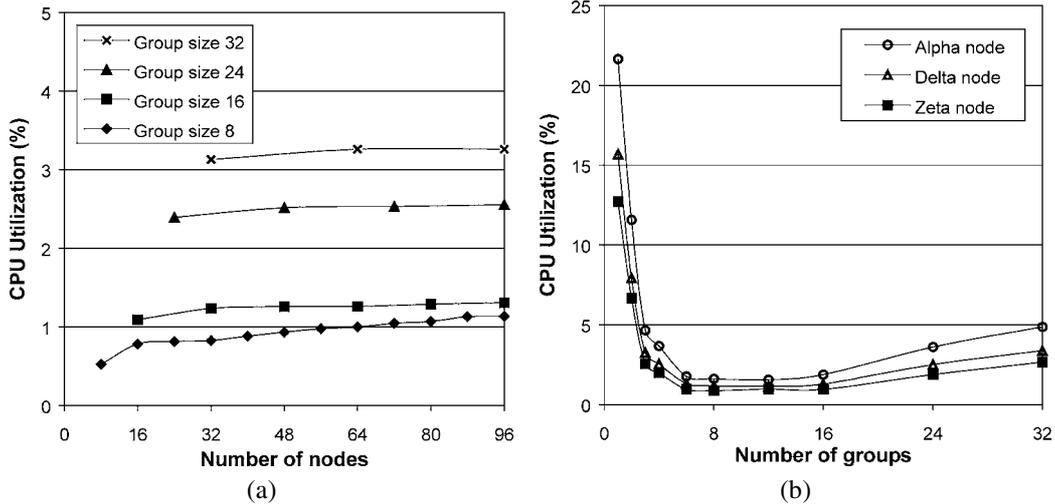


Figure 12. Scalability of CPU utilization in a layered system for several group sizes (a), and the impact of number of groups on CPU utilization in a layered system of 96 nodes (b).

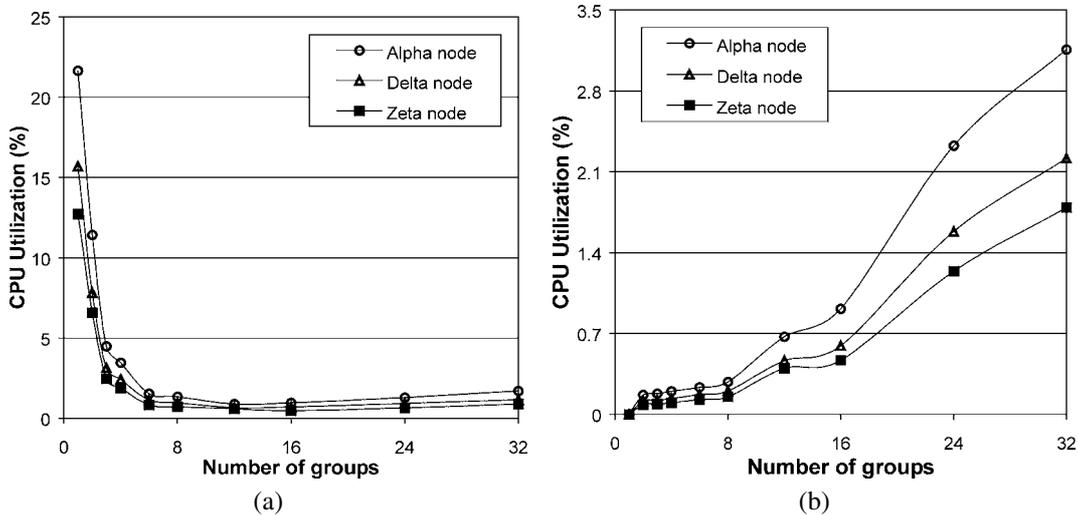


Figure 13. Impact of number of groups on CPU utilization in a layered system of 96 nodes: (a) L1 CPU utilization and (b) L2 CPU utilization.

consensus processing increases as  $O(g^2)$ , due to the increase in the size of the suspect matrix.

Finally, figure 14 presents a scalability comparison of flat and layered gossiping. In this case, flat gossiping uses RR and layered gossiping is set up as in the previous experiments. The group size for layered gossip is set to eight. It is observed that layered gossip scales well compared to flat gossip, with the CPU utilization per node for a 96-node layered system being approximately 6% that of a comparable flat system. On average, the flat system exhibits an  $O(n^2)$  scalability, while the layered system is virtually constant.

## 5. Resource utilization projections

Analytical modeling of performance of the failure detection and consensus service enables resource utilization projections for system sizes not available in the testbed. This section presents the performance projections for network utilization and processor utilization of the failure detection and consen-

sus service developed in this paper. Appropriate results are validated using the experimental results from the previous section.

### 5.1. Network utilization model and projections

Network utilization of the failure detection and consensus service is determined by the size of the gossip packet and the value of  $T_{\text{gossip}}$ . Figure 15 shows the packet structure for a flat system and a layered system with two layers. The gossip header field is 4 bytes in size and it specifies the type of the gossip packet and the length of the packet. A layer information unit ( $LIU_n$ ) contains the gossip information on the  $n$ th layer, specifically the bit-set vector, gossip list and the suspect matrix for that layer.

The bit-set vector is a bit vector whose  $i$ th bit is set to '1' if the  $i$ th group in the  $n$ th layer is alive, otherwise it is set to '0'. The length of the bit-set vector field is adjusted to an integral multiple of 8 to fit the byte-oriented structure of the

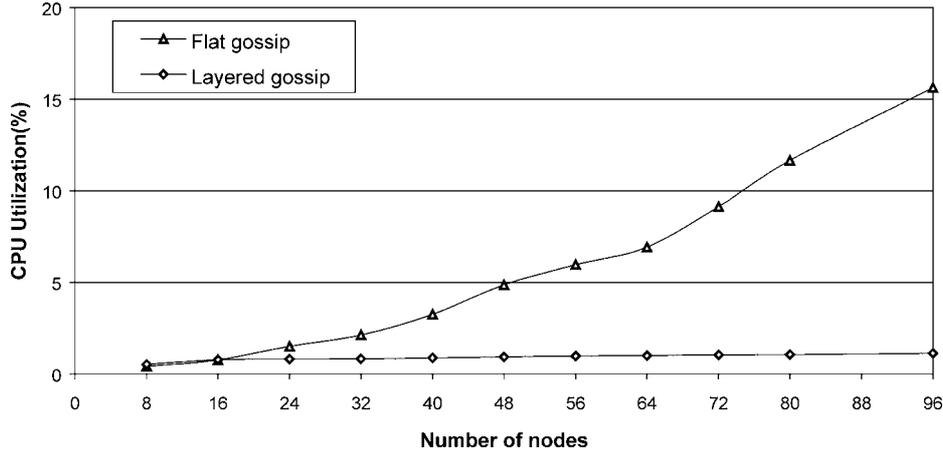


Figure 14. Comparison of scalability of CPU utilization in layered and flat systems.

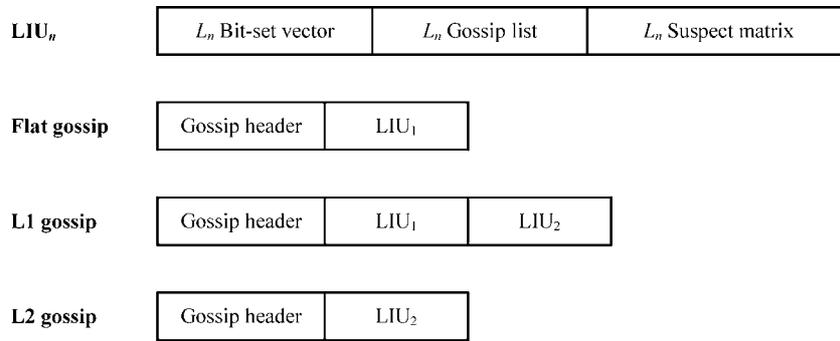


Figure 15. Packet structures of flat and layered gossip (two-layered) packets.

UDP packet. Information on only those groups that are alive is contained in the gossip list and the suspect matrix. The gossip list field is a sequence of bytes, with each byte containing ‘heartbeat’ data for each group in the  $n$ th layer. The suspect matrix field contains the  $n$ th-layer suspect matrix encoded into a bit sequence. The length of suspect matrix is also adjusted to fit into an integer number of bytes. As shown in figure 15, the  $i$ th-layer gossip packet in a  $n$ -layered system contains the gossip header followed by LIUs from the  $i$ th layer to  $n$ th layer.

Using  $n$  to represent the number of nodes in the system, the payload length of a gossip packet in bytes for a flat system is given by equation (1). The components in the equation include the length of the gossip header, the bit-set vector, the gossip list and the suspect matrix, respectively,

$$\begin{aligned} \text{payload length} &= 4 + \left\lceil \frac{n}{8} \right\rceil + n + n \left\lceil \frac{n}{8} \right\rceil \\ &= 3 + (n + 1) \left( \left\lceil \frac{n}{8} \right\rceil + 1 \right) \quad (\text{bytes}). \quad (1) \end{aligned}$$

The physical length of the gossip packet in the transmission frame is obtained by adding the overhead contributed by the UDP and Ethernet protocols (42 bytes) to the payload length. Thus, the gossip packet length is given by

$$\text{packet length} = 45 + (n + 1) \left( \left\lceil \frac{n}{8} \right\rceil + 1 \right) \quad (\text{bytes}). \quad (2)$$

Nodes send gossip packets every  $T_{\text{gossip}}$  seconds, thus the bandwidth utilization per node is given by

$$B_{\text{flat}} = \frac{45 + (n + 1) \left( \left\lceil \frac{n}{8} \right\rceil + 1 \right)}{T_{\text{gossip}}} \quad (\text{bytes/second}). \quad (3)$$

In figure 16(a), the network bandwidth per node predicted by the analytical model is compared with the experimental result. It is noted that the analytical results closely match the experimental results with a maximum prediction error of less than 0.2%. Also shown in the figure is the bandwidth per node projection for system sizes up to 256 nodes in a flat system. For example, by considering the cumulative effect of this per-node utilization, a 256-node system employing flat gossiping will consume an aggregate bandwidth of 1.8 GB/s, which is unacceptably large for even systems with high bandwidth availability.

For a layered system with  $l$  layers, where  $L_i$  denotes the length of the  $i$ th-layer gossip packet and  $f_i$  denotes the frequency of the  $i$ th-layer gossip, the bandwidth utilization per node is given by:

$$B_{\text{layered}} = \sum_i^l L_i f_i \quad (\text{bytes/second}). \quad (4)$$

Consider a two-layered system of  $g$  groups with each group containing  $m$  nodes, where  $n = mg$ . The expressions

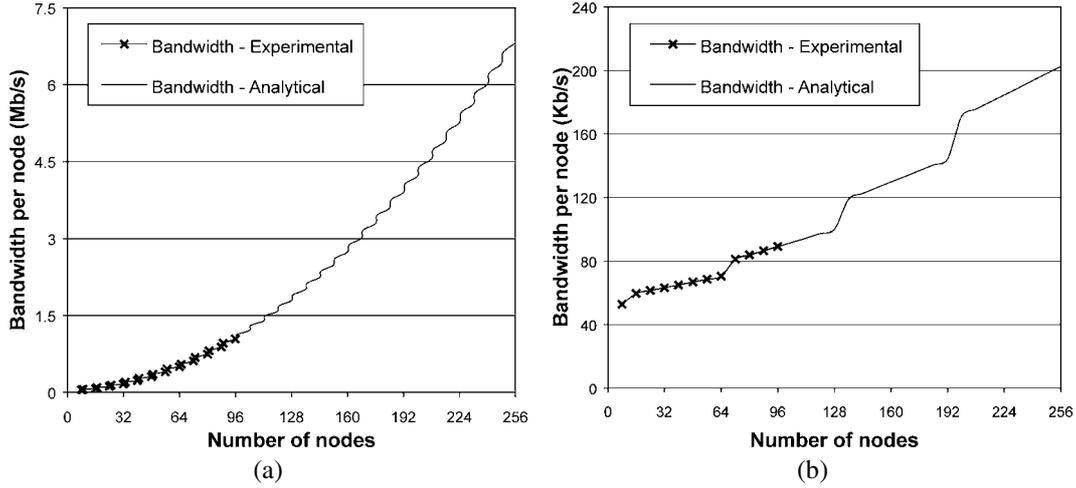


Figure 16. Validation of the analytical model for network bandwidth per node: (a) flat system and (b) two-layered system (group size = 8).

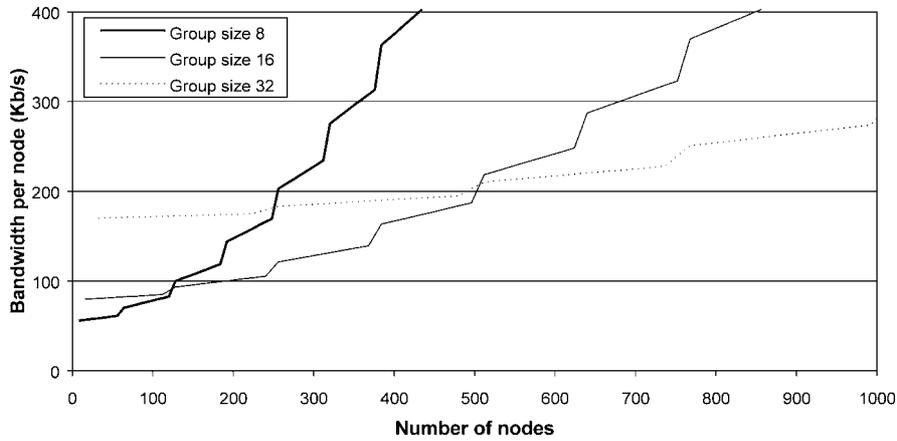


Figure 17. Bandwidth per node projection for a two-layered system for different group sizes.

for  $L_1$  and  $L_2$  obtained by extending the result from a flat system are given by:

$$L_1 = 44 + (m + 1) \left( \left\lceil \frac{m}{8} \right\rceil + 1 \right) + (g + 1) \left( \left\lceil \frac{g}{8} \right\rceil + 1 \right) \quad (\text{bytes}), \quad (5)$$

$$L_2 = 45 + (g + 1) \left( \left\lceil \frac{g}{8} \right\rceil + 1 \right) \quad (\text{bytes}). \quad (6)$$

Nodes in the first layer take turns in sending the second-layer gossip, hence each node sends second-layer gossip every  $mT_{\text{gossip}}$  seconds, while they send first-layer gossip every  $T_{\text{gossip}}$  seconds. Applying equation (4) to a two-layered system and noting that frequency of gossiping is the reciprocal of the time period, we derive the expression for bandwidth utilization per node of a two-layered system

$$B_{\text{two-layered}} = \frac{44 + (m + 1) \left( \left\lceil \frac{m}{8} \right\rceil + 1 \right) + (g + 1) \left( \left\lceil \frac{g}{8} \right\rceil + 1 \right)}{T_{\text{gossip}}} + \frac{45 + (g + 1) \left( \left\lceil \frac{g}{8} \right\rceil + 1 \right)}{mT_{\text{gossip}}} \quad (\text{bytes/second}). \quad (7)$$

In figure 16(b), the network bandwidth per node predicted by the analytical model for a two-layered system with a group size of 8 is compared with the experimental result. Here again the analytical results closely track the experimental results. Also shown in figure 16(b) is the bandwidth per node projection for system sizes up to 256. For example, by considering the cumulative effect of this per-node utilization, a 256-node system employing layered gossiping (with two layers and a group size of 8) will consume an aggregate bandwidth of 55 Mb/s, which is approximately 3% the utilization of the corresponding flat system.

Figure 17 shows the projection made by the analytical model for different group sizes. Bandwidth per node exhibits an overall second-order scalability. The crossover points between different curves determine the optimum group size for a given system size. For system sizes less than 128, a group size of 8 achieves the minimum bandwidth utilization while a group size of 16 is optimum in the range from 128 to 512. For system sizes larger than 512, a group size of 32 is observed to be optimum.

Equation (7) can be optimized under the constraint  $n = mg$  to yield an optimum value of  $B_{\text{two-layered}}$  for a given system size, thus yielding the expression for the optimum group size

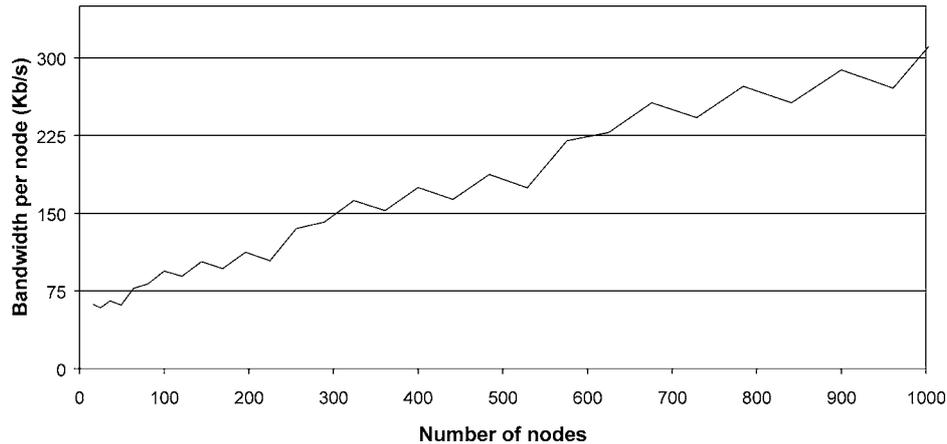


Figure 18. Bandwidth per node projection for a two-layered system ( $g = m = \sqrt{n}$ ).

in terms of  $n$ . Such an optimization is complicated by the discrete nature of the equation, and instead a simple heuristic solution is proposed. By choosing  $m$  and  $g$  to be equal to  $\sqrt{n}$ , it is observed that equation (7) yields a linear scalability in network utilization per node versus system size in a two-layered system. This heuristic is also supported by the experimental results presented in section 4.2.

Figure 18 shows the network utilization per node for a two-layered system obtained using the heuristic suggested above. Since the optimum network utilization per node calculated by actually solving the equation can be no worse than the heuristic, it follows that the optimum network utilization for a two-layered system is upper-bounded by the curve shown in the figure and thus achieves an overall  $O(n)$  scalability.

### 5.2. Processor utilization projections

Processor utilization of the failure detection and consensus service depends on several factors, which include the processor architecture, compiler version and the operating system details. Modeling such dependencies is not only complex but also has limited applicability. Instead, analytical projections are provided here for processor utilization on the three processors previously enumerated.

Figure 19 shows the variation of the processor utilization of a flat system with the system size, where the dotted lines show the experimental results and the solid lines show the values predicted by the curve fit. By checking the order of the computation loops in the implementation code, it can be concluded that the number of CPU cycles varies quadratically with the number of nodes for a flat system. It is observed that the resulting quadratic curve fit closely approximates the actual variation. The coefficients of the quadratic curve for the three architectures are shown in table 3.

In figure 20, the CPU utilization of a flat system is projected up to a system size of 256 nodes. It is observed that the maximum system size sustainable (i.e., where processor utilization is less than 100%) is approximately 250 nodes. However, since these same processors are responsible for more than just gossiping, processor utilization likely exceeds acceptable levels long before that threshold is reached.

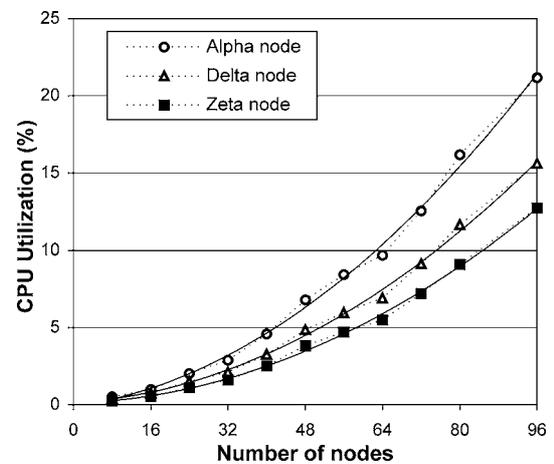


Figure 19. Processor utilization of a flat system with curve fit.

Table 3  
Coefficients of a quadratic curve fit for the processor utilization of a flat system.

Node	Processor utilization = $an^2 + bn + c$		
	$a$	$b$	$c$
Alpha	0.0019	0.0429	-0.001
Delta	0.0015	0.019	0.1336
Zeta	0.0013	0.0095	0.1042

As for layered systems, although a projection conducted in a similar manner to that provided for network utilization was not practical due to limits in the size of the available test-bed (e.g., an accurate curve fit and corresponding projection for a two-layered system with a group size of 16 would require a minimum of  $128 + 16$  nodes to observe at least two linear regions in the quadratic curve), the general trends can still be projected. To begin with, processor utilization of a layered system can be expected to scale in the same manner as its network utilization. Hence, we expect an overall  $O(g^2)$  scalability for a fixed group size, or equivalently an overall  $O(n)$  scalability when employing an optimal group size, with a profile similar to the sawtooth curve exhibited in figure 18.

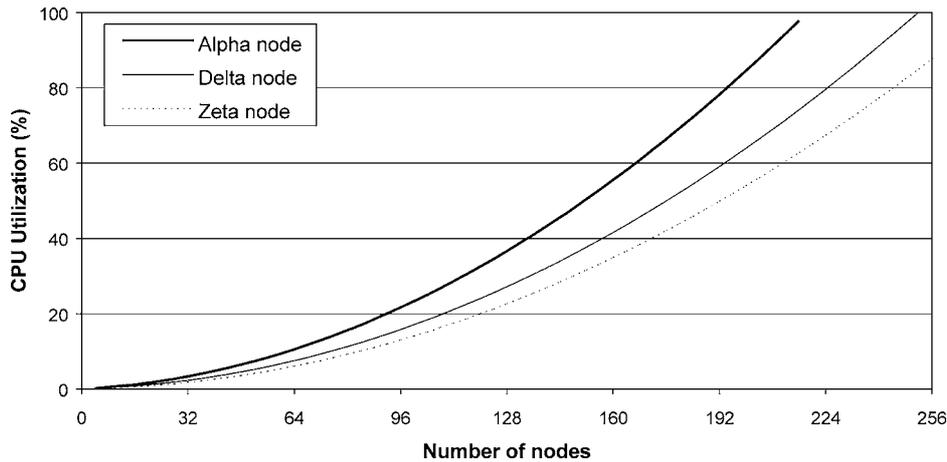


Figure 20. Processor utilization projection for a flat system.

Moreover, as was observed from the results in subsection 4.3, processor utilization in a two-layered system is insignificant ( $\sim 1\%$ ) for a system of approximately 100 nodes. Given the linear scalability of processor utilization versus system size for an optimal group size, and the modest slope exhibited in the experiments in the previous section, it can be projected that processor utilization will be less than 10% for systems up to 1000 nodes, which may or may not be considered significant depending upon the nature of the system.

Of course, when the system size should grow to the extent that processor utilization is considered significant, a logical next step would be the consideration of a third layer. As system size grows even further, perhaps even a fourth layer would be advantageous so as to maintain an insignificant level of utilization. Therefore, clearly there will exist crossover points in optimal processor utilization versus system and group size in terms of the number of layers employed. Such scalability and tradeoff comparisons involving schemes using three or more layers, in terms of resource utilization and consensus time, are left for future study.

## 6. Conclusions

In this paper, a comprehensive performance analysis of several system-wide schemes for gossip-style failure detection and consensus is presented based on experiments with a full implementation of the service on a cluster testbed of approximately 100 nodes. The performance and scalability of both flat and two-layered systems is compared and contrasted, with and without network-supported broadcast for consensus propagation, in terms of time to reach consensus, level of network bandwidth utilization, and level of processor utilization. Finally, a set of analytical models for network bandwidth utilization in flat and layered systems is developed, validated, and then employed to predict the level of resource utilization in systems of up to 1000 nodes.

With flat gossiping systems, consensus time is found to scale with system size following an  $O(n)$  trend. For a given system size and cleanup time, flat systems with consensus broadcast are found to provide a marginally lower consensus

time than flat systems that must rely on gossiping to propagate consensus. When system size exceeds 72 nodes, the random protocol for basic gossiping in a flat system outperforms protocols based on round-robin gossip since the latter require clock synchronization between nodes for optimal performance. In this work no effort was made to synchronize node clocks, thus exposing an inherent weakness in patterned gossiping.

In contrast with flat gossiping, a layered system exhibits superior scalability by providing consensus times that are virtually independent of system size at a magnitude that is significantly lower even for systems of intermediate size. For example, with a two-layered system of 96 nodes divided into groups of eight nodes each, and where support for consensus broadcast is available, the consensus time in a layered system is less than 70 ms or about 25% that of a comparable flat system. As the size of the groups in a layered system decreases, so too does the magnitude of the consensus time. Conversely, layered systems that must rely on gossip for consensus propagation are found to be the least scalable of all the schemes, with the consensus time increasing significantly with the system size. For instance, with a system of 96 nodes where support for consensus broadcast is not available, the consensus time approaches 1 s or approximately three times that of a comparable flat system with or without consensus broadcast when the group size was fixed at eight. However, employing fewer groups of larger size can mitigate this effect.

In a flat system, network utilization and processor utilization per node are both found to increase with an overall  $O(n^2)$  scalability. Conversely, a layered system is found to exhibit superior scalability in resource utilization versus system size, with a two-layered system exhibiting an overall  $O(g^2)$  scalability for a fixed group size, or equivalently an overall  $O(n)$  scalability when the number of groups and the number of nodes per group are reasonably balanced. For example, with a two-layered system of 96 nodes again divided into groups of 8 nodes each, each node in the system will consume a network bandwidth of approximately 90 Kb/s, which is only about 10% that of a comparable flat system. Similarly, processor utilization per node is only about 1% in the layered system

versus more than fifteen times that amount in a comparable flat system. Given the linear scalability of the resource utilization of a two-layered system and the low resource utilization observed up to a system size of 96, it is projected that a two-layered system can provide reasonably low resource utilization for system sizes approaching 1000 nodes. The use of more than two layers can of course be considered if the system size should grow to the extent that two layers are not sufficient to keep the resource utilization low.

Directions for future research include development and performance analysis of multi-layered systems employing three or more layers. Analytical models need to be developed that can project the performance of a multi-layered system for large system sizes. Another direction of interest is a focus on dependability analysis of the gossip service in terms of reliability, as well as number and type of faults tolerated, versus group and system size. The issue of fault-tolerant, distributed clock synchronization also needs to be addressed in support of gossip protocols based on round-robin scheduling. An efficient solution to this problem may use gossiping itself to distribute the time stamps of the nodes involved. Support for distributed clock synchronization would provide several benefits, such as the ability to better support deterministic protocols for larger system sizes, simplification and streamlining of a scheme for the insertion of new nodes, and as a bookmark for a fault-recovery journaling scheme. Finally, experimental studies at the application level are needed to evaluate the impact of the failure detection and consensus service on the performance of large-scale, distributed applications.

### Acknowledgements

The support provided by Sandia National Labs on contract LG-9271 is acknowledged and appreciated, as are equipment grants from Nortel Networks, Intel, and Dell that made this work possible.

### References

- [1] R. van Renesse, R. Minsky and M. Heyden, A gossip-style failure detection service, in: *Proc. of IFP International Conf. on Distributed Systems Platforms and Open Distributed Processing Middleware'98*, Lake District, England, 15–18 September 1998.
- [2] M. Burns, A. George and B. Wallace, Simulative performance analysis of gossip failure detection for scalable distributed systems, *Cluster Computing* 2(3) (1999) 207–217.
- [3] S. Ranganathan, A. George, R. Todd and M. Chidester, Gossip-style failure detection and distributed consensus for scalable heterogeneous clusters, *Cluster Computing* 4(3) (2001) 197–209.



**Krishnakanth Sistla** received the B.Tech. degree in Electrical Engineering from the Indian Institute of Technology, Madras in 1999, and the M.S. degree in electrical and computer engineering from the University of Florida in 2001. His research interests include fault-tolerant services for distributed systems, and microprocessor and system architecture with special emphasis on power and performance tradeoffs. He is currently working with the Enterprise Processors Group at Intel Corporation.  
E-mail: krishnakanth.v.sistla@intel.com.



**Alan D. George** is an Associate Professor of electrical and computer engineering at the University of Florida, and Director and Founder of the HCS Research Lab. He received the B.S. degree in computer science and the M.S. in computer-electrical engineering from the University of Central Florida, and the Ph.D. in computer science from Florida State University. Dr. George's research interests are in high-performance networks and architectures for parallel, distributed, and fault-tolerant systems and applications. He is a senior member of both the IEEE and the SCS.  
E-mail: george@hcs.ufl.edu.



**Robert W. Todd** received a B.S. degree in electrical engineering in 1995 and an M.S. degree in electrical engineering in 1996, both from Florida State University. His research interests include active networks and reconfigurable computing. He is currently a doctoral candidate in electrical and computer engineering at the University of Florida.  
E-mail: todd@hcs.ufl.edu.