Design, Performance and Dependability of a Peer-to-Peer Network supporting QoS for Mobile Code Applications

Amit Kapur, Natarajan Gautam
310, Leonhard Building
Industrial and Manufacturing Engineering
Penn State University
State College, PA 16801
e-m*ail*: amitkapur@psu.edu , ngautam@psu.edu

Richard Brooks
Applied Research Laboratory
Penn State University
State College, PA 16801
e-m*ail*: rrb5@psu.edu

Suresh Rai
Electrical and Computer Engineering
Louisiana State University
Baton Rouge, LA 70803
e-m*ail*: suresh@ece.lsu.edu

Abstract

With the advent of web applications such as Napster, Gnutella and Freenet, peer-to-peer (P2P) networks have gained unprecedented importance in the networking world. In this paper we study the effect of different network parameters on the performance of P2P networks supporting mobile code applications, and analyze various quality of service (QoS) issues such as response times, jitter and loss probability. We derive analytic expressions for (a) the number of hops using graph-theoretic techniques and, (b) the QoS measures using probabilistic models. For our studies, we have used three types of network models, namely Erdös-Rényi graph, small world graph, and scale free graph. Further, we formulate and solve an optimization problem to design the number of indexes and time out value, where an *index* refers to a database that maps mobile codes to nodes that contain them.

1 Introduction

This paper looks at the intersection of two current technologies for creating distributed adaptive systems: mobile code [9] and peer-to-peer (P2P) networking [12]. We are interested in creating a network infrastructure that is capable of adapting to malicious, possibly catastrophic events. Mobile code technology enables transmission and execution of programs between networked nodes. It supports adaptation by allowing nodes to reconfigure their software and change roles dynamically. P2P networks distinguish themselves from traditional client/server or master/slave networks in that there is neither a central point of control nor centralization of data. They could potentially support adaptation by allowing the network structure to evolve. To realize our goal of creating adaptive network services, we are creating a P2P infrastructure from existing mobile code daemons [3].

As described in [12], many technologies can be classified as P2P. The two most widely known implementations are Napster and Gnutella. Napster is a file-sharing network with only one central index. This index contains a database of users and their files. When a user connects to Napster, a list of available files on their machine is added to a central index. When the user requests a specific file, a list of participating machines containing the file is returned. The file can be retrieved from any machine on the list. This is an efficient architecture. File names and machine addresses contain tens of bytes. Files being exchanged contain megabytes of data. The large data transfers occur between machines chosen virtually at random. This tends to spread data traffic evenly throughout the Internet. On the other hand, its survivability is poor as a single failure or a court order can stop the entire network by switching off the central index.

Gnutella offers a radically different approach [7]. It is fully distributed with no single point of failure. Each node has an index of its own files. File discovery is performed by flooding the network with request packets. There appears to be serious scalability issues with this approach [Hong 2001, Ritter 2001]. Another application, Freenet [8] addresses this scalability issue. On the other hand, Gnutella has interesting survivability characteristics. To stop the Gnutella service, it would be necessary to stop every node on the Internet from running Gnutella.

This paper studies the question: what is an appropriate number of indexes for a P2P network? We analyze this problem in terms of Quality of Service (QoS), scalability, and survivability. Recall that Napster has 1 index. It is efficient, but has a single point of failure. Gnutella provides n indexes for n nodes. It, thus, lacks single points of failure, but does not scale well. We consider Napster and Gnutella as the extreme cases of a continuum. The number

of indexes could vary in the range of 1 to *n*. Before designing and implementing an optimal network, it is necessary that the network's performance and dependability be thoroughly analyzed. These two issues are critical in P2P networks because of their complex topologies and the unreliable environments they operate in. In P2P networks it is of interest to know how long it will take to retrieve a file and what portion of the requests are lost. These issues become more critical with each additional hop a request needs to travel and nodes going down due to random failures or attacks.

The layout of the paper is as follows: In Section 2, we provide some preliminary results such as network topology and hop count, which form the building block of our QoS analysis. The scenario under consideration is described in Section 3 with results for QoS measures calculated against different input parameters. Section 4 develops an analytical model for the QoS measures. We formulate and solve an optimization problem to select appropriate number of indexes and time-out value in Section 5. Section 6 concludes the paper and also gives the direction for future work.

2 Preliminaries

P2P networks are characterized by a lack of central authority. Implementations generally allow nodes to join and leave the infrastructure at will. We characterize the network as a graph consisting of *n* nodes and *a* bi-directional arcs connecting them. Each node has an associated degree *k*, which is the number of edges incident on the node. The length of each edge of the network is one, so if two nodes are connected by one edge then the shortest path between them is 1. We are interested in the general class of P2P networks and not a particular instance. For P2P systems, no central control exists to enforce a specific topology. In the case of Napster, file transfers are initiated at random among participants. These considerations support analysis of the problem with various graph formalisms [2].

2.1 Types of networks

In the following, we consider three classes of randomly generated graphs:

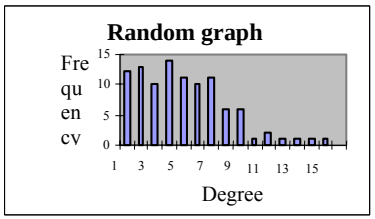
Random Graphs or Erdös-Rényi graphs – Consider a set of *n* nodes and a uniform probability *p* that an edge exists between any two nodes. The node degree distribution follows a Poisson distribution [2]. When these graphs have a single connected component, the average path length of these graphs grows as the logarithm of the number of nodes [1].

Small world graphs – They are a class of graphs with two properties: (*i*) average path length increases with the number of nodes in the same order of magnitude as random graphs, and (*ii*) there is a significant clustering of nodes (i.e. many nodes have multiple neighbors in common). For this class, we use the connected caveman model described in [14]. A set of fully connected components is constructed. One edge at random is rewired in each fully connected component so that the set of components is connected in a cycle. A small set of edges in the resulting structure is rewired at random. The node degree distribution depends on the number of edges re-wired.

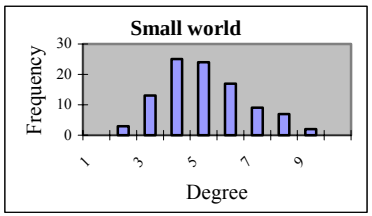
Scale-free graphs – A class of graphs where the probability *p (k)* that a node has degree *k* follows a power law distribution $p(k) \propto k^{-\gamma}$, where γ is a constant. Empirical studies have shown that many real-world systems including the Internet [1,4] and Gnutella [6] have this property. The average path length of scale-free networks grows more slowly with respect to the number of nodes than the path length of random networks.

The generality and applicability of these three models to many potential applications is attractive. They all scale well. The number of hops increases at most logarithmically with respect to the number of nodes. Due to the higher probability of having some nodes with very large degrees in scale-free networks, failure of those nodes could cause serious concerns while designing survivable systems. However, random (or Erdös-Rényi) graphs and small-world graphs that have very low probability of having nodes with high degrees are more suitable for survivable systems. Also, the clustered nature of small world graphs could be useful for detecting and containing system intrusions.

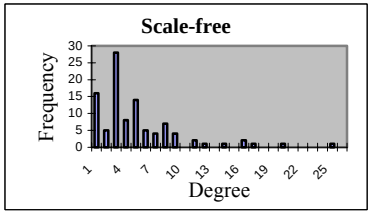
2.2 Comparison of the network types



(a)



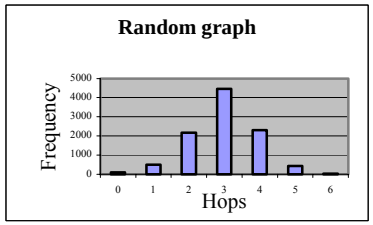
(b)



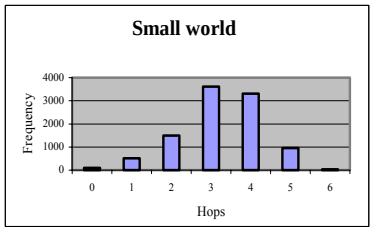
(c)

Figure 1: Histograms of degree. (a) Random graph. (b) Small world. (c) Scale-free.

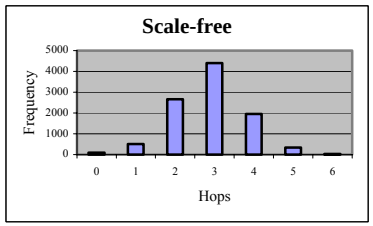
We now compare the three types of networks: Random graph, small world and scale-free. We considered networks with 100 nodes and average degree of 5. In Figures 1 and 2, we illustrate histograms of the degree and number of hops, respectively, for the three network types. From these histograms it can be seen that the variance in degree of nodes is highest for scale-free and lowest for small world. Also, there is not a very significant difference in the mean and the variance of the number of hops for all the three network models.



(a)



(b)



(c)

Figure 2: Histogram of hops. (a) Random graph. (b) Small world. (c) Scale-free.

2.3 Analytical model of hop count

We would like to analytically investigate the hop count characteristics against the number of nodes, n, average degree, k_{ave}, and network type. A simple estimator for the average number of hops (l_{ave}) between any two nodes in a random graph is [Dorogovtsec 2001]:

$$l_{ave} \sim ln\,[n] / ln\,[k_{ave}]$$

Empirical results from [14] indicate that the relationship between graph size and average number of hops is similar for random and small world graphs. Scale-free graphs appear to grow even more slowly [1]. To analyze further, we define the *clustering coefficient*, which expresses the cliquishness of the network (C = 1 for a complete graph, C = 0 for a tree). From [11]:

*C = 3 * (Number of triangles in network) / (Number of connected triples of vertices)*

*C = 6 * (Number of triangles in network) / (Number of paths of length two)*

As an example, the clustering coefficient C for 1-lattice is given as

$$C = \frac{3}{4} \times \left(\frac{k-2}{k-1} \right) + \frac{i(i+1)}{k(k-1)} \qquad (1)$$

$$\text{where } i = \begin{cases} \frac{3k}{2} - n + 1; & k+1 \le n \le \frac{3k}{2} \\ 0 & ; \; n > \frac{3k}{2} \end{cases}$$

The expression for C contains an extra $\frac{i(i+1)}{k(k-1)}$ term that applies to all nodes n such that $k+1 \le n \le \frac{3k}{2}$. Thus, if k = 100 then the expressions in [14] fails to give correct value for the clustering coefficient C in 1-lattice for nodes $101<n<151$. The expression (1) helps fix the anomaly. Also in [11] there is a factor *mutuality M* for the mean number of paths of length two leading to the nodes two hops away (squares in the network):

$$M = ave\ [k / [1 + C^2 (k-1)]] / k_{ave}$$

As noted in [11], mutuality M provides the density of "squares" as against clustering coefficient C, which refers to the "fraction of transitive triples" in the network. Mathematically

$$M = \frac{\text{mean number of vertices two steps away, } N_2}{\text{mean paths of length two to those vertices, } P_2} \quad (2)$$

In 1-lattice network, we have from equation (2)

$$M = \begin{cases} 0; & n = k+1 \\ \frac{1}{2k-n+2}; & k+1 < n < 2k \\ \frac{4(k-1)}{k(k+2)}; & n = 2k \\ \frac{4}{k+2}; & n > 2k \end{cases} \quad (3)$$

Best estimator of $q_2 = M\ (1\text{-}C) * (Average\ [k^2] - Average\ [k])$, where q_2 is the expected number of nodes 2 hops away from a node. M accounts for the possible over count of nodes two hops away relying solely on C. [11] provides a derivation using generating functions approach, which shows that the number of nodes h hops away can be approximated using only two factors: the number of nodes one hop away and the number of nodes two hops away. Results are approximated since detailed graph structures may cause specific graph instances to deviate, and they assume single fully connected component in the graph. According to them (z_i = mean number of nodes h hops away):

$$z_m = (z_2 / z_1)^{m-h} * z_1$$

Based on the above literature study, we now derive a simple estimate of the number of nodes reachable after exactly h hops as:

$$q_h = (k_{ave} - 1) * q_{h}\text{-}1\ (1\text{-}C) * M\ [h] \quad \text{and} \quad q_1 = k_{ave}$$

$$For\ h > 1\ q_h = (1\text{-}C)^{h-1} * (k_{ave} - 1)^{h-1}\ k_{ave} \prod_{i=2}^{h} M[h]$$

The average number of nodes reachable after one hop is by definition k_{ave}. For the second hop, every node reached by one hop has $(k_{ave} - 1)$ degrees free. C of those $k_{ave} * (k_{ave} - 1)$ degrees are the percentage on the average already reachable by one hop. $M\ [2]$ compensates for the over count due to quadrilaterals. $M\ [h]$ generalizes this. A more accurate estimate of the number of nodes reachable in h hops can be given by taking into account the greater likelihood of connecting to a node with a higher degree [11] (again $h > 1$, for $h = 1$ it remains the same):

$$q_h = (1\text{-}C)^{h-1} \prod_{i=2}^{h} M[h] \sum_{k=1}^{n-1} (k-1)^{h-1} * k * p_{kJ}$$

This would yield the expected number of hops between any two nodes chosen at random as:

$$\alpha(n, a, s) = (0 + q1 + 2\ (q2 - q1) + \ldots) / n$$

$$= (1 / n)\{ k_{ave} + 2\ [(1 - C)\ M[2]\ (\sum_{k=1}^{n-1} (k-1)^{h-1}\ k\ p_k) - k_{ave}\ ]$$

$$+ \sum_{h=3}^{maxhops} h\ \{[(1 - C)^{h-1} \prod_{i=2}^{h} M[h] (\sum_{k=1}^{n-1} (k-1)^{h-1}\ k\ pk)] -$$

$$[(1 - C)^{h-2} \prod_{i=2}^{h} M[h] (\sum_{k=1}^{n-1} (k-1)^{h-1} k\ p_k)]\}\}$$

where α (n, a, s) is the expected number of hops for n nodes, a arcs and s standard deviation of degree.

3 Scenario: Platform for Simulation and Analytical models for QoS

3.1 Problem Description

This paper addresses the issue of finding out the optimum number of indexes that should be there in a P2P network. Each index carries with it the information regarding who in its vicinity has what document or file. The various inputs that will be used to study the QoS issues are number of nodes n, number of arcs a, type of network (random graph, small world, and scale-free), communication speeds (link speeds) l, processing speeds d, number of different documents (modules) m, and number of replicas of documents c.

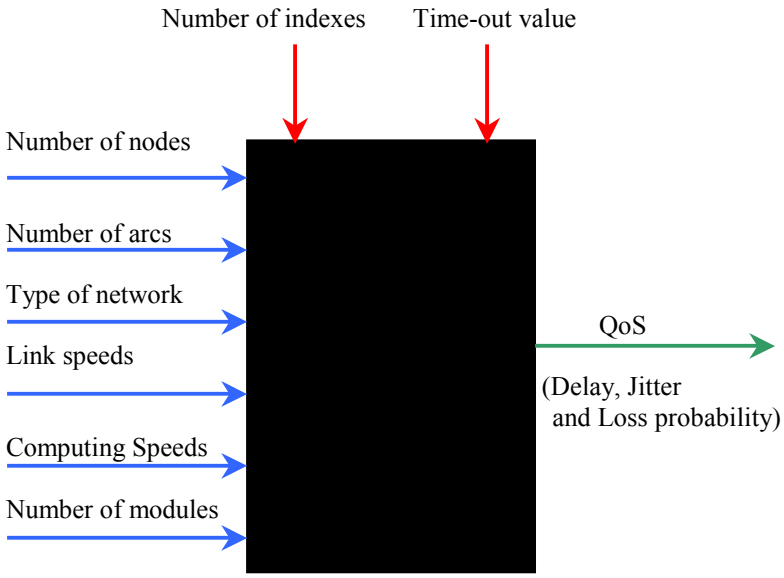


Figure 3: Problem description

We also assume that the nodes go up and down from time to time independent of other nodes. Nodes go down due to several reasons including denial-of-service attacks. The QoS issues considered include: *average response time, jitter* and *loss probability*.

We consider a scenario where a node (called the source node) makes a request for a document and starts a timer. If the source node does not receive the document before time θ_t (called the time-out value), the request is dropped. We compare the effect of varying the number of indices (I ranging from1 to n) and the time-out value, θ_t, on the QoS of the three networks types. Figure 3 illustrates the variables that are in our control to design (top of the box) and that are not in our control (left of the box).

3.2 Request-response process

If we divide the entire network into as many groups as there are number of indices each index node has a database of only those nodes that are in its group. The entire request-response process is described in Figure 4.

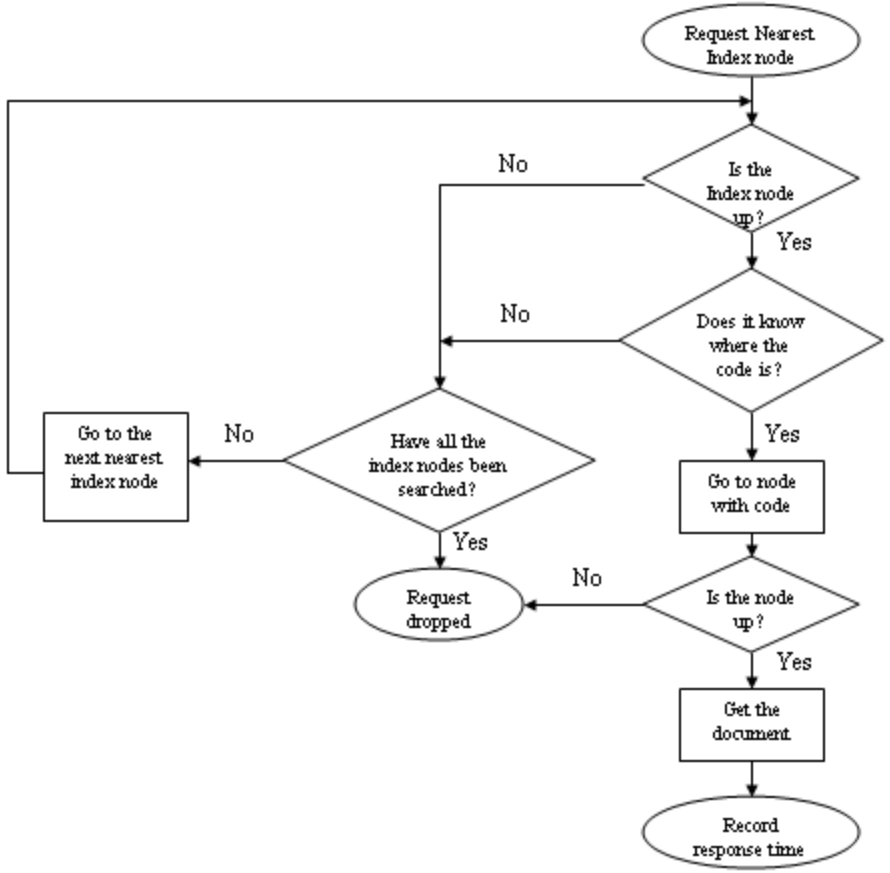


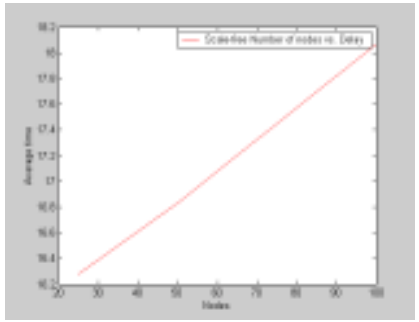
Figure 4: Flowchart of request-response process

3.3 Preliminary Results

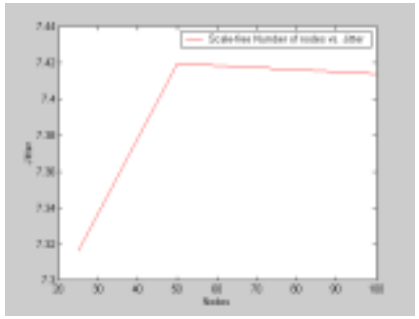
Several simulations were run to study the effect of varying input parameters on the three QoS issues. It was observed that all the three network types (Random graph, Small world, and Scale-free) reacted similar to the variations in the parameters. The parameters that were varied (Figure 3) were: number of nodes, average arc degree, number of indices I, number of copies of a document co, the time-out value and the probability of nodes being up p. We considered our base model with n = 25, k_{ave} = 3, I = 1, m = 25000, co = 1, p =

0.95 and infinite time-out value, θ_{inf}. The reason for choosing one index and infinite time-out is that it then resembles the structure in Napster. The results for varying different input parameters are summarized in Figures 5-10. Due to space restrictions only Scale-free network results are depicted with the understanding that the other networks, Random graph and Small world, do not produce significantly different results.

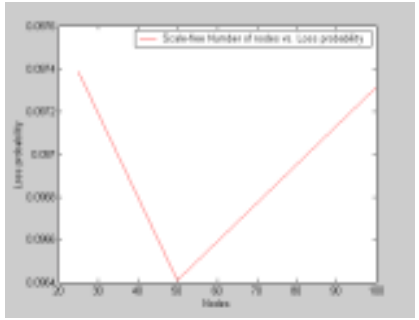
➢ *Varying the number of nodes:* Figure 5 shows the effect of increasing the number of nodes from 25 to 50 and 100 on the QoS. Notice from Figure 5 that the delay, jitter and loss probability do not change significantly as the number of nodes increase. This shows that the network is very scalable. The reason for the scalability is that with nodes the number of edges also increases in the same proportion. However the loss probability will increase if the time-out value is small (Figure 9 c).



(a)



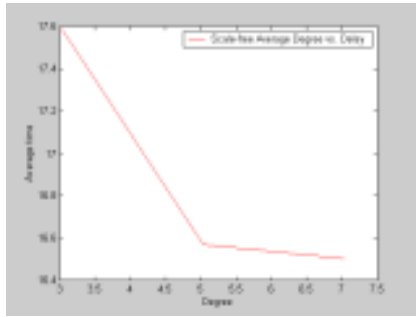
(b)



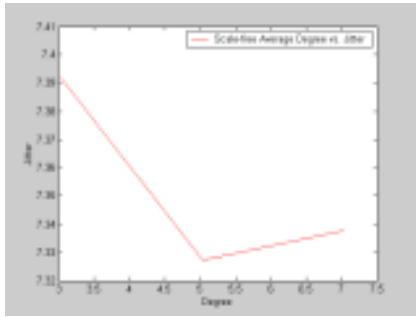
(c)

Figure 5: Scale-free QoS. (a) Delay vs. Number of nodes. (b) Jitter vs. Number of nodes. (c) Loss probability vs. Number of nodes

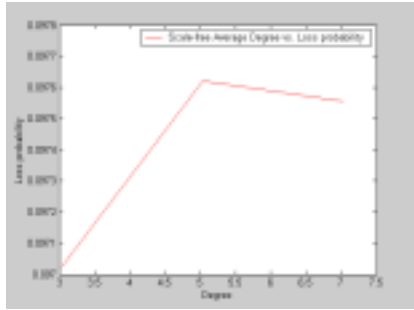
➢ *Varying the average arc degree:* We compared the QoS for three average arc degrees: 3, 5 and 7. The results are in Figure 6. Increasing the average arc degree means that on an average the nodes will have more edges and hence the number of hops from one node to another will decrease. Decrease in hops means that it will take less time for a source node to get to the index node and the destination node. Hence the *delay* and the *jitter* will drop with increasing average degree. The *loss probability* will only be affected if there is a finite time-out value. In that case the loss probability will decrease with average arc degree.



(a)



(b)

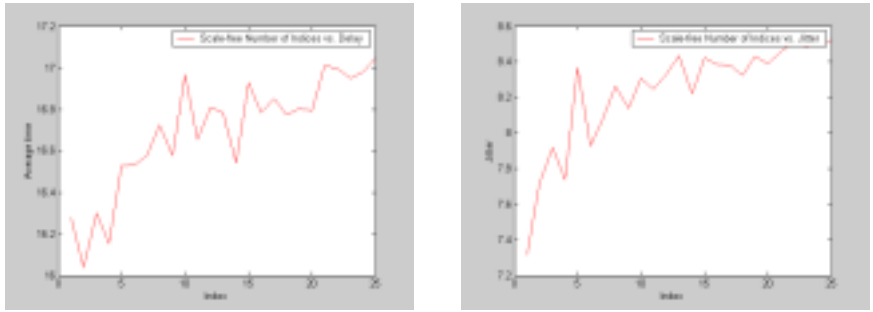


(c)

Figure 6: Scale-free QoS. (a) Delay vs. Average degree. (b) Jitter vs. Average degree. (c) Loss probability vs. Average degree.

➢ *Varying the number of index nodes:* We considered several indexing schemes from a centralized (Napster) to a fully distributed (Gnutella)

mechanism. Figure 7 shows the results that were obtained on varying the number of indices. Intuitively d*elay* and *jitter* will increase with number of indices, *I*, as the time to search a document increases with *I*. The *loss probability* will remain more or less the same if there is only one copy of the document as nodes fail independent of each other. But if we increased the probability of an index node being up as the number of nodes increases, the loss probability decreases as shown in Figure 7(c).

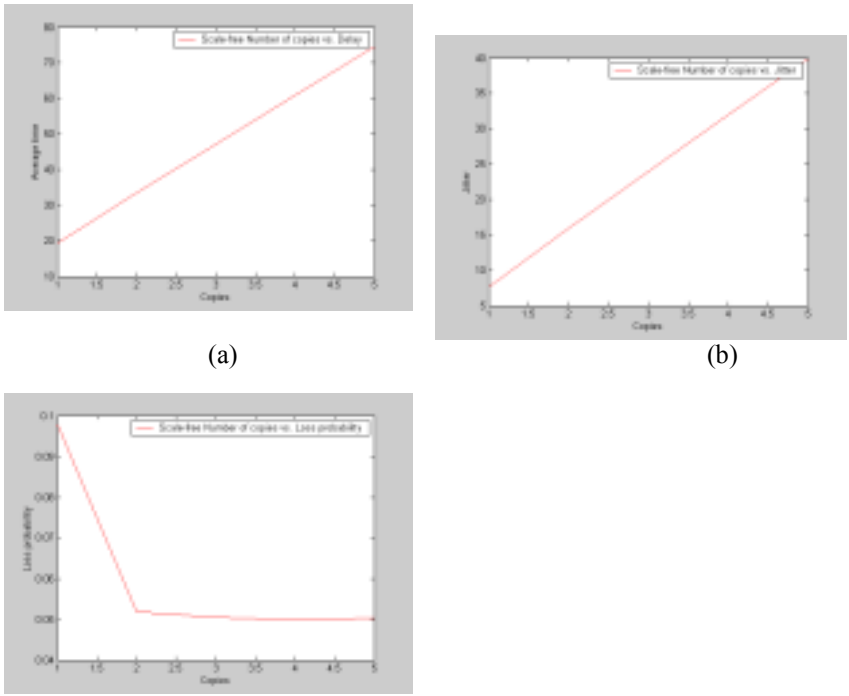


(a) (b)

(c)

Figure 7: Scale-free QoS. (a) Delay vs. Number of indices. (b) Jitter vs. Number of indices. (c) Loss probability vs. Number of indices.

➢ *Varying the number of copies:* Figure 8 shows how increasing the number of copies influences the QoS measures. Increasing the number of copies means that even if an index node or destination node with a particular document is down, the source node might get it from some other node (provided the other index node and destination node are up). With more number of copies, the time to fulfill a request increases as all the indexes are searched. Hence, the average *time* and *jitter* increase with number of copies but the *probability of loosing* a request drops down.

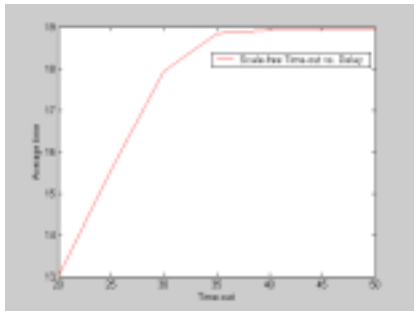


(a) (b)

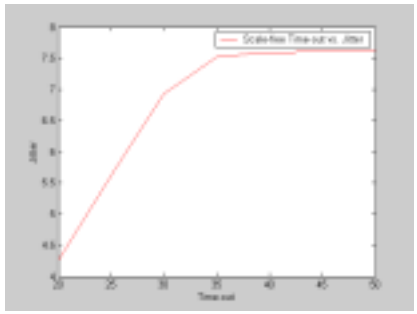
(c)

Figure 8: Scale-free QoS. (a) Delay vs. Number of copies. (b) Jitter vs. Number of copies. (c) Loss probability vs. Number of copies.

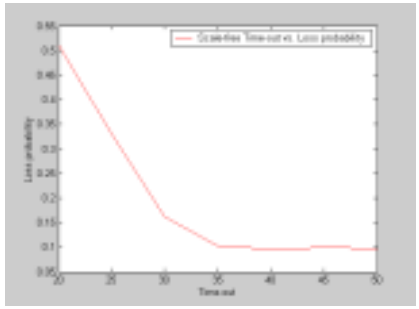
➢ *Varying the time-out value:* When a source makes a request, it sets a time out value which if it expires, the source assumes the request is lost. Decreasing the time-out value from there would decrease the delay and jitter as they are measured for only requests that are completed. On the other hand, the loss probability will increase on decreasing the time-out value, as more requests would be dropped. Figure 9 shows the effect of time-out value on the three QoS measures.



(a)



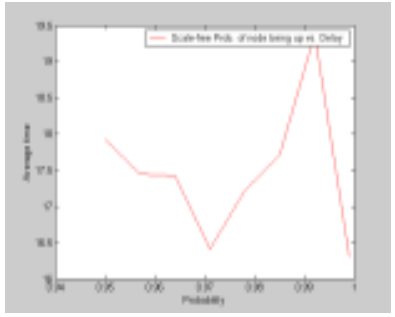
(b)

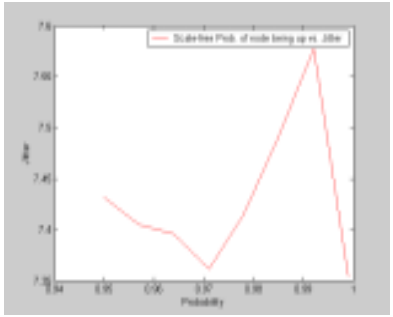


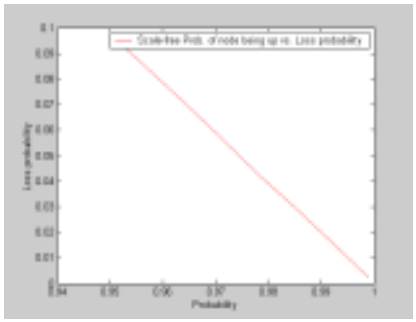
(c)

Figure 9: Scale-free QoS. (a) Delay vs. Time-out value. (b) Jitter vs. Time-out value. (c) Loss probability vs. Time-out value.

➢ *Varying the probability of nodes being up:* Increasing the probability of nodes being up will affect only the percentage of requests being lost. The loss probability decreases if the nodes are up for a longer time. However, increasing the probability of nodes being up does not affect the delay and the jitter. The affect of varying the probability on QoS measures is summarized in Figure 10.







(a)

(b)

(c)

Figure 10: Scale-free QoS. (a) Delay vs. Probability of nodes being up. (b) Jitter vs. Probability of nodes being up. (c) Loss probability vs. Probability of nodes being up.

4 Analytical Models

Since the number of copies, number of indexes and the time-out values are under our control, we would like to select them in an optimal manner. In order to do that, we first build analytical models that relate QoS measures to the design parameters. We initially assumed time-out is infinite and then use an appropriate model to include time-outs. The analytical models are then compared (in Section 5) with the simulation results.

4.1 QoS measures: Infinite time out

(a) The expected delay in retrieving a document, *E(T)* is (see Appendix 1 for definitions and analysis):

$$E(T) = (1/((1-(1-p)^{co})\,p))\{2q\,l \sum_{i=1}^{I} \mu_i(n,a,s)\,pi + pdmco/I \sum_{i=1}^{I} p_i(i-1) + 0.5dmco(i/n + 1/I) \sum_{i=1}^{I} p_i + (q\,l + E(B)\,l) \sum_{i=1}^{I} \alpha_i(n,a,s)\,pi\}.$$

where

p_i = *P (Code location is found in index I) For i = 1, 2, …,I*

$$= \sum_{k=1}^{i} \binom{i-1}{k-1} * \binom{I-c}{k-1} / \binom{I}{k-1} * p^k * (1-p)^{i-k}$$

$$* \min[co/I-k+1),1]$$

(with the understanding that $\binom{(i)}{(k)} = 0$ if $i < k$) and

p_0 = *Probability that all the indices with the code are down*

$$= 1 - \sum_{j=1}^{I} p_j = 1-(1-p)^{co}$$

(b) The variance in delay (which is the square of the *jitter*) is given by

$$Var\ (T) = E\ (T^2) - \{E\ (T)\}^2$$

where the expression for $E\ (T^2)$ can be obtained from Appendix 2.

(c) Proportion of requests lost or loss probability

$$L = 1 - p\ [1 - (1-p)^{co}]$$

4.2 QoS measures: Finite time-out

The above three models assume that the time-out value is infinite. This means that a request is lost only if the index node or the destination node with the document is down. If the time-out value was finite, then requests would also be lost if the time to get a response exceeds the time-out value. Since the delay is a sum of a large number of independent random variables, we can approximate using central limit theorem that $T \sim$ Normal [E (T), Var (T)]. Now the probability that time-out occurs even when the document is available will be

$$P\ (T > \theta) = \varepsilon = 1 - \Phi\ [(\theta - E\ (T)) / \sqrt{Var\ (T)}]$$

Then, the response time given that document is retrieved before time-out will be

$$E\ (T_\theta) = \int_{-\infty}^{\theta} x\ f\ (x)\ dx\ /\ (1 - \varepsilon)$$

where $f(x)$ is the normal probability distribution function. The variance will be

$$Var\ (T_\theta) = \{(1 - \varepsilon) \int_{-\infty}^{\theta} x^2\ f\ (x)\ dx - \varepsilon\ [E\ (T_\theta)]^2\} / (1 - \varepsilon)^2$$

The loss probability will be $L_\theta = 1 - p\ [1 - (1-p)^{co}](1 - \varepsilon)$

5 Results

5.1 Comparison of analytical and simulation models

The *delay*, *jitter* and *loss probability* for the three networks were determined using both the analytical model as well as simulation models to verify the results. For all the 3 networks, the number of nodes are 100, average arc degree as 3.00 and the number of indices as 4. Also 10 replications were taken for all the simulation runs (this comparison is for infinite time-out value without considering node failure). The results are summarized in Table 1(for 1 copy) and Table 2 (for 3 copies):

Type of Network	**Analytical Model**			**Simulation Model**		
	Delay	*Jitter*	*Loss prob.*	*Delay*	*Jitter*	*Loss prob.*
Random Graph	18.9582	5.7169	0.1477	17.0031	7.4065	0.2141
Scale-free	17.8171	5.9683	0.1279	16.8547	7.3984	0.1652
Small-world	21.9941	4.5980	0.2108	17.4444	7.6001	0.2917

Table 1: Analytical and Simulation results for 1 copy

Type of Network	**Analytical Model**			**Simulation Model**		
	Delay	*Jitter*	*Loss prob.*	*Delay*	*Jitter*	*Loss prob.*
Random Graph	20.8487	7.6610	0.3947	15.6902	7.2839	0.3236
Scale-free	19.4134	7.9690	0.3522	15.2349	7.1359	0.2876
Small-world	22.3077	7.9237	0.4554	15.6423	7.4876	0.3481

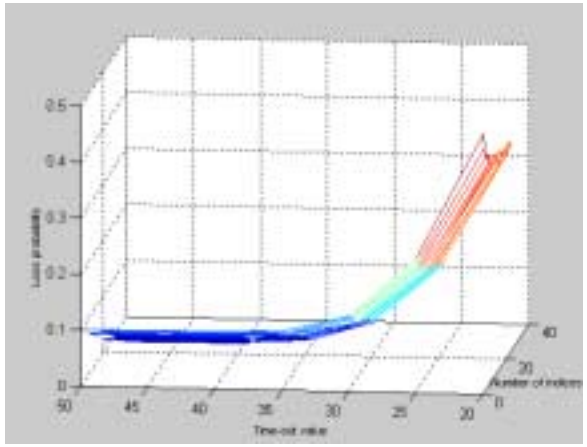
Table 2: Analytical and Simulation results for 3 copies

From the above two tables it can be seen that results from analytical and simulation models are fairly close. The difference in the two results is due to two approximations that we made in deriving the analytical models. One approximation is the Central Limit Theorem we used to incorporate finite time-out value and the other approximation we made was for the mean and variance of the number of hops.

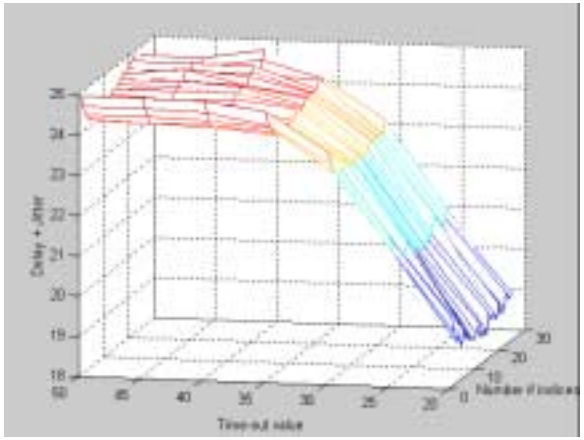
5.2 Optimal number of indices and time-out value

Out of the three controllable input parameters that we have, we chose to optimize only the number of indices and the time-out value but not the number of copies. The number of copies can be decided based on the storage capacity of the P2P network. For our objective we used only 1 copy of a document. The number of indices was varied from 1 to n (centralized to completely distributed network) to determine the optimal number of indices in a P2P network. Also the time-out value was varied from infinite to a certain value so that not more that 10% of the requests were lost. The optimization problem can be written in terms of θ *and* I, (the decision variables) as:

Minimize:	*Delay* (T_θ) + *Jitter* $(\sqrt{Var}\ (T_\theta))$
Subject to:	*Loss probability* $(L_\theta) \leq 10\%$
	Time-out $(\theta) < \infty$
	Number of indexes $(I) \geq 1$



(a)



(b)

Figure 11: The optimal solution. (a) Delay + Jitter vs. Number of indices and Time-out value. (b) Loss probability vs. Number of indices and Time-out value.

In our scenario, there is a trade-off between loss probability and delay+jitter. With number of indexes and time-out value, the loss probability decreases and the average time increases. The optimal value for indexes and time-out for our set of input parameters were 9 indexes and 30 time units respectively. Nine indexes are better than having only one as now there is not a single point of failure in the P2P network. It is also better than having all the nodes as indexes as the traffic along the network will not be high. The time-out value of 30 is neither too large nor too small.

6 Conclusion and Future work

We studied the effect of various input parameters (such as number of nodes, number of copies, number of arcs, number of indices, time-out value, and probability of node being up) on three different network types: Random graph, Small world and Scale-free. We found that the QoS measures varied significantly only with the inputs and not with the network type. We derived analytical expression for the number of hops in a network. We used the mean and variance in the number of hops in a stochastic model to derive the QoS measures. These were used in a mathematical programming problem to determine the optimal number of indexes and time-out value in a P2P network for an acceptable loss probability and minimum response time plus jitter.

In this paper we considered a scenario where only one request was processed at a time. But in reality the number of requests per second is usually more than one. This complicates the problem further, but it would be interesting to study how the P2P network would behave in such a situation. Another interesting issue would be to do a cost analysis of a P2P network. Cost of storing and maintaining data in the network would play an important role when determining the optimum number of indexes.

Acknowledgement and Disclaimer

For Drs. Brooks and Gautam, this material is based upon work supported by the Office of Naval Research under Award No.N00014-01-1-0859. Any opinions, findings, and conclusions in this publication are those of the authors and do not necessarily reflect the views of the Office of Naval Research. For Dr. Rai, this material is based upon work supported in part by US Army Research Office under Award No.C-DAAD19 01-1-0646 and NSF grant CCR 0073429. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the author and does not necessarily reflect the view of the sponsoring agencies.

References:

[1] R. Albert, and A. -L. Barabási, "Statistical Mechanics of complex networks," arXiv: cond-mat/0106096v1, June 2001.
[2] B. Bollobás, "Random Graphs", Cambridge University Press, Cambridge, UK, 2001.
[3] R. R. Brooks, E. Grele, W. Kliemkiwicz, J. Moore, C. Griffin, B. Kovak, and J. Koch "Reactive Sensor Networks: Mobile Code Support for Autonomous Sensor Networks," Distributed Autonomous Robotic Systems DARS 2000, Pp. 471-472. Springer Verlag, Tokyo, October 2000.
[4] S. N. Dorogovtsev, and J. F. F. Mendes, "Evolution of Networks," arxiv: cond-mat/0106144v2, September 2001.
[5] T. Hong, "Chapter 14: Performance," in Peer-to-Peer Harnessing the Power of Disruptive Technologies, A. Oram, ed. pp. 203-241, O'Reilly, Beijing, 2001.
[6] M. Jovanovic, "Modeling large-scale peer-to-peer networks and a case study of Gnutella," Master's thesis, University of Cincinnati, 2001.
[7] G. Kan, "Chapter 8: Gnutella" in Peer-to-Peer Harnessing the Power of Disruptive Technologies, A. Oram, ed. pp. 94-122, O'Reilly, Beijing, 2001.
[8] A. Langley, "Freenet", pp. 123 – 132b, Orielly 'Peer-to-Peer', 2001
[9] D. Milojičić, F. Douglis, and R. Wheeler, "Mobility: Processes, Computers, and Agents", ACM Press, Reading, MA, 1999.
[10] M. E. J. Newmann, S. H. Strogatz, and D. J. Watts, "Random Graphs with arbitrary degree distributions and their applications," arXiv: cond-mat/007235, May 7, 2001.
[11] M. E. J. Newmann, "Ego-centered networks and the ripple effect or Why all your friends are weird," Working Papers, Santa Fe Institute, Santa Fe, NM, 2001 http://www.santafe.edu/sfi/publications/workingpapers/01-11-066.pdf.
[12] A. Oram, ed. Peer-to-Peer Harnessing the Power of Disruptive Technologies, O'Reilly, Beijing, 2001.
[13] J. Ritter, "Why Gnutella Can't Scale. No, Really", 2001 http://www.darkridge.com/~jpr5/doc/gnutella.html
[14] D. J. Watts, Small Worlds, Princeton University Press, Princeton, NJ, 1999.

Appendix 1: Analytical expression for delay

The analytical expression for delay is determined by conditioning on where the document is, using the information in index nodes, and computing the time to reach the document and retrieving it. The expected delay would be:

$$E(T) = \sum_{i=1}^{I} E(T/\text{ document available in index } i) * P(\text{document available in index } i).$$

$$= (1/((1-(1-p)^{co})\,p)) \sum_{i=1}^{I} \{2E(\text{Time to travel from source to index } i) + E(\text{time to search } (I-1) \text{ indexes, some possibly down}) + E(\text{Time to find code in } i^{th} \text{ index})\}\, p_i$$

$$+(1/((1-(1-p)^{c})\,p)) \sum_{i=1}^{I} \{E(\text{Time required to reach destination node in region } i) + E(\text{Time to search node in region } i) + E(\text{Time to respond})\}\, p_i.$$

$$E(T) = (1/((1-(1-p)^{co})\,p)) \{ \sum_{i=1}^{I} [2\mu_i(n, a, s)\, qlpi + p(i-1)\, dmco/Ip_i + 0.5dmco/Ip_i + \alpha_i(n, a, s)\, qlpi + 0.5dmco/np_i + \alpha_i(n, a, s)\, E(B)\, l\, p_i]\}.$$

Summing over will yield the result. Note that $\alpha_i (n, a, s)$ = Average number of hops from a node to i^{th} farthest index (or a node in that region) when there are *n* nodes, *a* arcs and *s* standard deviation in the network. This is obtained from the analytical model of hop count described in Section 2.3.

$$\mu_i (n, a, s) = \sum_{k=1}^{i} \alpha_k (n, a, s)$$

q = Average query size (usually a query size is 60 bytes).
B = Average code size (100 Kb).

Special Cases:

- If co = 1 and p = 1 (Number of copies is one and nodes do not fail)
  $p_i = 1 / I$ for $1 \le i \le I$ and $p_0 = 0$
- If co = 1 (Number of copies is one)
  $p_i = p / I$ for $1 \le i \le I$ and $p_0 = 1\text{-}p$
- If p = 1 (Nodes do not fail),
  for $j \le I - c + 1$, $p_j = {}^{(I-co)}C_{(j-1)} * {}^{1}C_{(j-1)} * co/I\text{-}j+1$
  for $j > I - co + 1$ and $j = 0$, $p_j = 0$

Appendix 2: Analytical model for jitter

Jitter is defined as the square root of the variance of *T*, the response time. The variance is calculated similar to *E (T)* by conditioning on the location of the document's index.

$$Var (T) = E (T^2) - \{E (T)\}^2$$

$$E (T^2) = (1 / ((1-(1-p)^{co}) p)) \sum_{i=1}^{I} E (T^2 / \text{document available in the vicinity of index } i)\, p_i$$

$$E (T^2 / \text{document available in the vicinity of index } i) = Variance (T^2 / \text{document available in the vicinity of index } i) + E (T / \text{document available in the vicinity of index } i)^2$$

$$E (T / \text{document available in the vicinity of index } i) = \{2ql\mu_i (n, a, s) + p (i-1) d m co / I + 0.5 d m c / I + \alpha_i (n, a, s) q l + 0.5 d mco/n + \alpha_i (n, a, s) E (B) l\}. = E (Ti)$$

$$Var (T_i) = Variance (T^2 / \text{document available in the vicinity of index } i)$$

In order to complete the derivation, we need some notations.

Let *Hi* be the number of hops to a node in region i.

$$E (H_i) = \alpha_i (n, a, s)$$
$$Var (H_i) = \beta_i^2 (n, a, s)$$
$$Let\ Y_i = H_1 + H_2 + H_3 + \ldots H_i$$

Therefore,

$$E (Y_i) = \mu_i (n, a, s)$$

$$Variance (Y_i) = \sigma_i^2 (n, a, s) = \sum_{j=1}^{i} \beta_j^2 (n, a, s)$$

Variance (T_i) will be

$$Var (T_i) = 4q^2 l^2 \sigma_i^2 (n, a, s) + (d\ mco/I)^2 (I-1)\, p\, (1-p) + 1/12 (dmco/I)^2 + \beta_i^2 (n, a, s)\, q2\, l^2 + 0.5 (dmco/n)^2 + l^2 \{Var (B) \beta_i^2 (n, a, s) + Var (B) \alpha_i^2 (n, a, s) + E (B)\, 2\beta_i^2 (n, a, s)\}$$

Therefore,

$$E (T^2) = (1 / ((1-(1-p)^{co})\, p)) \sum_{j=1}^{i} \{Var (T_i) + [E (T_i)]^2\}\, pi$$

and

$$Var (T) = E (T^2) - \{E (T)\}^2$$

Biography

Amit Kapur is a Graduate Student in the Harold and Inge Marcus Department of Industrial and Manufacturing at the Pennsylvania State University. He received his Bachelors degree in Mechanical Engineering from the University of Pune, India. He is currently pursuing his MS in Industrial Engineering and Operations Research at Penn State University. His research involves Quality of Service analysis and optimal design of peer-to-peer networks by building and testing analytical models against simulations.

Dr. Gautam is an Assistant Professor in the Harold and Inge Marcus Department of Industrial and Manufacturing Engineering at the Pennsylvania State University. He received his B.Tech. degree in Mechanical Engineering from the Indian Institute of Technology, Madras, and his M.S. and Ph.D. in Operations Research from the University of North Carolina at Chapel Hill. He is a member of IEEE, INFORMS and MAA, and a senior member of IIE. He is an Associate Editor for the INFORMS Journal on Computing and the Newsletter Editor as well as Website Editor for the INFORMS Applied Probability Society. His research interests are in the areas of modeling, analysis and performance evaluation of computer, telecommunication and information systems.

Dr. Brooks is head of the Distributed Systems Department of the Applied Research Laboratory of the Pennsylvania State University. His areas of research expertise include: sensor networks, critical infrastructure protection, mobile code, and emergent behaviors.

Dr. Brooks received a Ph. D. in Computer Science from Louisiana State University in 1996. He has a B.A. in Mathematical Sciences from the Johns Hopkins University, and performed

graduate studies in computer science and operations research at the *Conservatoire National des Arts et Metiers* in Paris, France.

His work experience includes being Manager of Systems and Applications Programming for Radio Free Europe / Radio Liberty in Munich, Germany. Consulting tasks Dr. Brooks has performed include: the implementation of a stock trading network for the French stock exchange authority, and expansion of the World Bank's internal computer network to Africa and the Former Soviet Union.

Dr. Rai is a Professor with the Department of Electrical and Computer Engineering at Louisiana State University, Baton Rouge, Louisiana. Dr. Rai has taught and researched in the area of reliability engineering, fault diagnosis, parallel and distributed processing, Internet, and ATM. He is a co-author of the book *Wave Shaping and Digital Circuits*, and tutorial texts *Distributed Computing Network Reliability* and *Advances in Distributed System Reliability;* last two published from IEEE Computer Society Press. He has guest edited a special issue of IEEE Transactions on Reliability on the topic *Reliability of Parallel and Distributed Computing Networks*. For last 11 years, he is working as an Associate Editor for *IEEE Transactions on Reliability*. Dr. Rai has worked as program committee member for several international conferences.

Dr. Rai has published about 100 technical papers in the refereed journals and conference proceedings. He received the best paper award at the 1998 *IEEE International Performance, Computing, & Communication Conference* (Feb. 16-18, Tempe, Arizona; paper title: S. Rai and Y. C. Oh, Analyzing packetized voice and video traffic in an ATM multiplexer).

Dr. Rai is a senior member of the IEEE and member of the ACM.