Analysis of a Local Computer Network with Workstations and X Terminals

E. Drakopoulos
AT&T Bell Laboratories
Naperville, IL, 60566

Abstract

In this paper we study the performance of a network based computing system that consists of a number of clients and a server, interconnected by a local area network. Each client represents a diskless or swapful workstation, or an X terminal, and generates application workload using either a conventional UNIX file system or the Network File System (NFS). To characterize the application workload and the effect of the protocol (login, rlogin, NFS), the user interface (workstation, X terminal) and its configuration (memory size, disk storage), we perform a series of measurement experiments in a controlled environment. During each experiment the traffic on the network, and the activity on the various system resources are monitored. The measurements obtained from the experiments are used to parameterize a queueing network model of the system, which is then used to project the performance of the system under various load conditions, to identify system bottlenecks, and to compare design alternatives.

1 Introduction

In this paper we study the performance of a network-based computing system that consists of a number of clients and a server, interconnected by a carrier-sense multiple-access with collision detection (CSMA/CD) local area network. Each client represents a diskless or swapful workstation, or an X terminal. The workload is generated by highly interactive users, who use software development tools for structured software design and analysis.

Previous work related to the subject of workload and performance modeling of network-based distributed computing systems, is presented in[1–5], and the references therein. In[1], measurement experiments and modeling are used to study the performance of single-user workstations that access files remotely over a local area network. In[2] several service alternatives for storage systems for client–server style distributed systems, are compared. A modeling methodology for evaluating the performance of distributed multiple computer systems, is presented in[3]. The approach uses queueing network theory and graph models of program behaviour, to obtain an estimate of the average execution time of a parallel implementation of a program in a distributed environment. Finally, a flexible parametric distribution–driven synthetic workload model, that can be calibrated to reflect the characteristics of the workload found in any NFS environment, is presented in[4].

In this paper, to characterize the application workload in the network-based distributed system under study, we performed a series of measurement experiments in a controlled environment. During each experiment a benchmark script that was designed to represent typical user operations, was executed in an otherwise idle system. Each benchmark session was executed under the X window system, for the following four cases:

(a) NFS workload: A user logs in a workstation and executes the benchmark using the Network File System (NFS) to access all necessary data files from the server.

(b) Login workload: A user on a X terminal uses login to access the server, and then executes the benchmark under the server's local UNIX Filesystem.

(c) Rlogin workload: A user on a workstation uses rlogin to access the server, and then executes the benchmark under the server's local UNIX Filesystem.

(d) Login/NFS workload: A user on a X terminal uses login to access a workstation, and then executes the benchmark accessing all necessary data files from the server, using NFS (Login/NFS workload).

During the experiments, the activity on the various system resources (server CPU, storage devices, client CPU), and the traffic on the network were monitored. Specifically, a network monitoring tool (tcpdump) was used to collect information for each packet transferred over the network. The traffic measurements were analyzed to compute the distribution of packet sizes and packet interarrival times.

To project the performance of the system under load, and to compare design alternatives, we use the measurements obtained during the experiments to parameterize a queueing network performance model. The model is used to study the performance of the
system under various load conditions, to identify system bottlenecks by considering the utilization of system resources, and to determine the maximum number of active clients that the system can support with an acceptable response time. A client is said to be active if he/she generates load to the system, and idle otherwise. The application workload is highly interactive, and therefore an active client spends a significant amount of time in thinking mode; in other words, an active client does not constantly submit back to back commands for execution in the system.

The organization of the paper follows. Section 2 describes the workload characterization. The experimental testbed is described in Subsection 2.1, the application benchmark in Subsection 2.2, and the traffic characteristics of the various workloads in Subsections 2.2.1-2.2.3. Section 3 presents the queuing network model. In Section 4 we use the model to study the performance of the system for various load conditions. Finally, Section 5 summarizes the paper.

2 Workload characterization

The workload is generated by software developers who use Computer-Aided Software Engineering (CASE) tools for structured software design and analysis. To quantify the workload generated by a typical user, we ran a series of controlled experiments with an application benchmark. The benchmark was designed to represent typical operations performed by a user. Next we describe the experimental testbed and the application benchmark.

2.1 Experimental Testbed

The testbed for our experiments consisted of:

- A SUN 4/470 server, with 64 MB RAM, three 3 MB/s IPI disk drives, and one 6 MB/s IPI disk drive. A NFS accelerator was also used in some of our experiments.

- Ten SUN SPARCstation SLCs with 8 MB or 16 MB RAM. Both diskless and swapful workstation configurations were considered.

- One X terminal (AT&T 730X+) with 4MB RAM, that supports the X Window System.

One of the workstations (configured with 16 MB RAM and a 600 MB SCSI disk drive) was used to run tcpdump in order to monitor the traffic on the network during the experiments. The server and the clients were interconnected by a AT&T STARLAN-10 which is a 10 Mb/s CSMA/CD local area network, compatible with the IEEE 802.3 10base5 specification, and Ethernet. The server and the workstations ran the SunOS 4.0.3, and the X terminals MIT X11R4. The experimental testbed is shown in Figure 1.

2.2 Application benchmark

The test consists of a series of controlled experiments with an application benchmark. During each experiment a single user executes the benchmark script on an otherwise idle system. The activities of the user during the experiment were obtained by an accounting package (acctcom) that reads process accounting files and provides for each process among other parameters, the CPU consumption (in user and system mode), the number of blocks read and written, the mean memory size used, and the elapsed time. System performance data were obtained by standard SunOS commands such as vmstat, netstat, iostat, nsfstat.

The application benchmark script was designed to represent typical operations performed by a software developer involved in Computer Aided Software Engineering. The software development tool used in our experiments provides an integrated multi-user environment that supports user-extensible applications build upon template driven tools. The application benchmark was executed under the X window system in four different modes, each associated with a different workload type:

- A user logs in a workstation and executes the benchmark using the Network File System (NFS) to access all necessary data files from the server (NFS workload).

- A user on a X terminal uses login to access the server, and then executes the benchmark under the server's local UNIX Filesystem (Login workload).

- A user on a workstation uses rlogin to access the server, and then executes the benchmark under the server's local UNIX Filesystem (Rlogin workload).

- A user on a X terminal uses login to access a workstation, and then executes the benchmark on the workstation accessing all necessary data files from the server, using NFS (Login/NFS workload).

To characterize the workload generated by each benchmark session, we measured the following parameters:

1. Number of commands executed per session: The workload generated by each benchmark session, is characterized by a number of commands, which during their execution require service at the various system resources;

2. Sum of command elapsed times: For each command, the elapsed time is defined as the time interval between submission to the system and completion;

3. Length of measurement interval: The measurement interval is equal to the time that we monitor the performance of the system during the benchmark session;

4. Session system response time: The session system response time, is equal to the sum of the residence times at the various system resources (service and waiting time) for all the commands executed during the benchmark session;
5. **Client CPU consumption**: Total client CPU service required by all commands executed during the benchmark session;

6. **Server CPU consumption**: Total server CPU service required by all commands executed during the benchmark session;

7. **Number of I/O requests per session**: Total number of physical disk I/O requests by all commands executed during the benchmark session;

8. **Disk service time per I/O request**;

9. **Size, timestamp, and protocol** of each packet transferred over the network during each session. The traffic measurements were analyzed to compute the distribution of the packet size, packet interarrival time, protocol type, and to study the network utilization.

Next we discuss the traffic measurements associated with each workload type.

### 2.2.1 NFS workload

In Figures 2a–2c we show typical network traffic statistics for the experiments were nine users on workstations execute the application benchmark using NFS to access the necessary data files from the server. The workstations are diskless SUN SPARCstation SLCs, configured with 16 MB of memory. In Figure 2a we plot the network utilization over 1 sec intervals (the point zero in the time axis corresponds to the starting time of the experiment). Notice from the figure that the network utilization is high before the beginning of the experiment due to the necessary information that must be transferred to the workstations from the server (application loading phase). After this initial phase the network utilization is relatively low with significant spikes at the points where data exchange takes place between the workstations and the server.

The histogram of the packet size distribution (data file and a 14 Byte addressing header) is shown in Figure 2b. Among all the packets in the network about 32% are TCP packets with sizes mainly 60 and 98 bytes. The majority of these packets are associated with Xterm connections in the X window system, and represent traffic generated by active applications such as `zclock`, and `zshipt` that a typical user maintains. The remaining 68% of the packets are UDP packets. The sizes of the UDP packets (NFS traffic) are mainly 138, 146 and 1514 bytes. The average packet size is 330 bytes, the median 138, the mode 60, and the standard deviation 494.

The histogram of the packet interarrival time is shown in Figure 2c. Packet interarrival time is defined as the difference between the time when the transmission of two successive packets began. The average packet interarrival time is 9.8 msec, the median 1.8 msec, the mode 1.2 msec, and the minimum 0.36 msec. The skewness of the distribution is equal to 7, and the coefficient of variation 2.4, indicating a skewed to the right distribution with a very heavy tail. This means that most of the packets arrive very close to each other (within 1–5 msec) but there are some packets with relatively large time intervals between them.

Next we study the effect of the workstation configuration on the traffic characteristics. For this purpose, in Figure 3a, we show the network utilization for a typical experiment where a single user executes the application benchmark on a diskless workstation configured with 16 MB RAM. The network utilization for a typical experiment where a single user executes the application benchmark on a workstation configured with 16 MB of memory and 100 MB of local disk space used for swapping and temporary file storage, is shown in Figure 3b. By comparing this figure with Figure 3a that corresponds to the same experiment but with diskless workstation configuration, we note a significant reduction in the network utilization after the application loading phase. This difference is due to the fact that by using a swap disk on the client side we reduce significantly the number of requests that have to go to the server.

To study the tradeoffs between client memory expansion and client disk availability we repeated the above experiment using a workstation configured with 8 MB memory. The network utilization is shown in Figure 3c. Notice that the network utilization is very high, even higher than the network utilization observed in the case of 16 MB diskless workstation configuration. The above indicates that for the application under consideration the workstation configuration with 8 MB of memory and 100 MB of local swap disk is inefficient. This is probably due to the fact that the 8 MB memory size is too small for the particular application (the application requires at least 10 MB of memory).

### 2.2.2 Login/Rlogin workload

Next we analyze the data collected during the experiments where one user on a X terminal executes the application benchmark on the server. The network statistics are shown in Figures 4a, b, and c. Notice that the network utilization is significantly lower compared to that of the NFS workload. The packet rate is higher but the bit rate is lower, due to the fact that the login workload traffic consists of a large number of small packets. The average packet size for this experiment is 179 bytes, the median 36, the mode 86, and the standard deviation 262. The histogram of the packet interarrival time distribution is shown Figure 4c.

The traffic statistics for the rlogin workload are similar to that of the login workload and therefore are not shown here. In the case of rlogin workload the average packet size is 204 bytes, the median 98 and the mode 86.

### 2.2.3 Login/NFS workload

The network statistics for the experiment where one user on a X terminal uses login to access a diskless
workstation and then executes the application benchmark via NFS, are shown in Figures 5a and 5b. Notice that the network traffic and the packet size distribution combine the characteristics of both NFS and login workloads. The average packet size is 380 bytes, the median 138, the mode 86, and the standard deviation 523.

3 Queueing network model
To project the performance of the system under load, and to compare design alternatives, we use the results of our experiments to parameterize a queueing network performance model of the system. This model is used to study the performance of the system under various load conditions, and to identify system bottlenecks by considering the utilization of system resources. Using the queueing network model, we identify the maximum number of active clients (a client is said to be active if he/she generates load to the system) that the system can support with an acceptable response time.

The system under study consists of a number of clients (diskless or swapful workstations, and/or X terminals) that share a file server over a CSMA/CD local area network. Each client is modeled by a fixed rate service center with processor sharing (PS) service discipline, that represents the client CPU. The file server is modeled as a collection of queueing centers that represent hardware resources. The CPU is modeled by a fixed rate service center with PS service discipline, and the disk subsystem by a number of fixed rate first come first serve (FCFS) service centers, each representing an individual magnetic disk. A separate customer chain is used in the model to represent the workload generated by each client node. To model the user think time we use an infinite server service center. All service times are assumed to be exponentially distributed.

The local area network is modeled by an infinite server service center, with service time equal to the average time \( E(f_j) \) a customer \( j \) spends in the network. Each customer \( j \) arrival process to the network is a batch arrival process, with average batch size equal to \( E(n_{pj}) \) packets. To derive an expression for the average time \( E(f_j) \) a customer \( j \) spends in the network, we assume that the customer batch arrival process is Poisson, and that the batch sizes are independent and identically distributed, with a geometric distribution. In this case, it has been shown that the customer batch delay distribution is identical to the packet delay distribution.

Let \( \lambda_j \) denote the arrival rate to the network for the customer \( j \) batch arrival process. Then the arrival rate of the customer \( j \) packet arrival process to the network is equal to \( \lambda_j E(n_{pj}) \). The assumption of geometrically distributed batch sizes makes the customer \( j \) packet arrival process, a renewal process, i.e., the successive packet interarrival times are independent and identically distributed. For our analysis, we assume that the packet interarrival times are exponentially distributed, which implies that the packet arrival process follows a Poisson distribution. Note that the Poisson model is an approximation to the measured packet arrival process, but the fact that the measured packet arrival process is characterized by packet interarrival times that correspond to almost full packet transmission times, tends to keep the actual number of collisions and protocol delays, smaller than predicted by the Poisson model.

The average time \( E(f_j) \) spend by customer chain \( j \) in the network, is equal to the sum of the average packet waiting time \( E(w_p) \), the average transmission time \( E(n_{pj}) E(b) \), where \( E(b) \) denotes the packet transmission time which is defined as the ratio of the packet size to the channel transmission speed, and the average propagation delay \( \tau_p/2 \) between sender and receiver. Hence,

\[
E(f_j) = E(w_p) + E(n_{pj}) E(b) + \tau_p/2.
\]

The average packet transmission time \( E(b) \) is computed from the packet size distribution obtained from the traffic analysis described in Section 2. Expressions for the average packet waiting time \( E(w_p) \), and the average transmission time \( E(n_{pj}) E(b) \) are given in reference [5]. The queueing network model for diskless clients is shown in Figure 6. In the case of a swapful client a FCFS service center with exponential service time is used to model the local swap disk.

The workload generated by each workstation is characterized by commands which during their execution require service at the various system resources. A separate customer chain is used to model the commands submitted by each client. The measurements obtained through the experiments are used to parameterize the queueing network model.

The model consists of two submodels, the network, and the computing subsystems (clients and server), and is solved by using an iterative algorithm similar to the one presented in [8]. The basic steps of the solution algorithm are the following:

Step 1: Set the arrival rates \( \lambda_j \) to the local area network to some initial values.

Step 2: Check if the total packet arrival rate to the network is less than \( 1/(E(b) + \tau_p + 2\tau_p) \) (stability condition). If not, modify the arrival rates to meet the stability condition. Calculate \( E(f_j) \) using (1).

Step 3: Solve the queueing network model by using Mean Value Analysis.

Steps (2) and (3) of the algorithm are repeated until the following convergence criterion is satisfied:

\[
\Delta_n = \sum_{j=1}^{N} (\lambda_j^n - \lambda_j^{n-1}) < 10^{-6}
\]

where \( \lambda_j^n \) and \( \lambda_j^{n-1} \) denote the chain \( j \) arrival rates at the LAN for the \( n^{th} \) and the \( (n-1)^{th} \) iterations, respectively.
4 Results

In this section we use the model to study the performance of the system for various load conditions. The performance measures of interest are:

1. Application benchmark response time, defined as the sum of the residence time (service and waiting time) at each system resource.
2. Throughput, defined as the number of command completions per unit time.
3. Utilization of the shared system resources (server resources, and communication network).

The model developed in this paper is used to compute the mean values of system performance measures, to identify system bottlenecks by considering the utilization of system resources, to compare various design alternatives, and to determine the maximum number of active clients that can supported without violating system performance requirements.

The performance measures predicted by the model, were validated against actual measurements, and were found to be within 5–10% of the measured system response times, and 1–2% of the measured system throughput, and device utilizations. We note, that the projections made by the model, are most useful when viewed as relative values for comparing the effects of changes in workload intensity, or system design.

In Figure 7 we plot the system response time as a function of the number of users in the system for various workload types and system configurations. Notice that the number of users that can be supported with acceptable response time in the case of rlogin/login workload is small, due to the heavy server CPU requirements that characterize this workload. This is illustrated in Figure 8 where we show the server CPU utilization as a function of number of users in the system. In the case of NFS, the response time is higher at low system load compared to the response time in the case of rlogin/login, due to the larger overhead involved. But as the load increases the response time for rlogin/login increases rapidly due to the contention for the server CPU, while on the other hand NFS appears to be much more robust due to the decentralization of the compute load.

From the performance curves associated with the login/NFS mode of operation (where a user on a X terminal logs on a workstation and runs the application from there using NFS) we note that the login/NFS mode of operation can significantly increase the capacity of the system and provide acceptable performance to users with X terminals on their desks. We estimate that a properly configured workstation (with sufficient memory and local swap disk) can support around three X terminals for the application workload under consideration. In this case the login/NFS mode of operation presents a cost effective solution for supporting a large number of users.

In Figure 7 we also show the response time curves for swapful client configuration. Notice the performance improvement compared to the diskless case, which results from the fact that with the local swap disk the amount of requests can not be satisfied locally (i.e., they need to go to the server) is significantly reduced.

Finally in the same figure we show the response time curve associated with a configuration where a NFS accelerator is installed in the server. The NFS accelerator is a VME-bus board that improves NFS write performance and random file system access performance by using a 1 MB non-volatile memory as a write cache for synchronous disk I/O. The device intercepts synchronous write requests to disks and stores the data to its non-volatile memory. The cache is flushed to the disks asynchronously in large groups, which enables the disk drive controllers to optimize the order of writes to the disks, and hence increase disk efficiency.

As the response time curve in Figure 7 shows, the use of the NFS accelerator results in significant performance improvement. To further study its impact, in Figures 9 a, and b, we show the performance of the system as a function of the number of NFS users when there are 5 active rlogin users (background load). Specifically, Figure 9a shows the average utilization of the shared system resources (server CPU, server disks, and network), for server configurations with (dotted curves) and without (solid curves) the NFS accelerator; Figure 9b shows the response time as a function of the number of active NFS users in the system, for server configurations with (dotted curve) and without (solid curves) the NFS accelerator. Notice that without the NFS accelerator the response time of the NFS users is significantly higher than that of the rlogin users until the load increases to a point where the contention for the server CPU degrades significantly the performance of the rlogin users. With the NFS accelerator installed the crosspoint between the two curves is located at significantly lower load.

The use of the VME-bus NFS accelerator results in a reduction of the disk utilization (see Figure 9a), but it also increases the load on the server CPU because of the overhead due to data copying between the main memory and the accelerator on-board memory. In general the use of the NFS accelerator will increase the capacity of the server, but for environments with very heavy NFS load the stream of writes to the accelerator board may become the system bottleneck. Furthermore, in this case the load on the server CPU is usually very high and therefore the system cannot handle the extra load imposed by the VME bus based NFS accelerator overhead operations. Under these conditions, the use of the accelerator board is not recommended.

5 Summary

In this paper we studied the performance of a network based computing system that consists of a number of clients and a server, interconnected by a local area network. The application workload is highly interactive, and it is characterized by a series of measurement experiments in a controlled environment. The measurements were used to parameterize a queueing network model of the system, which was then used to project the performance of the system under var-
ious load conditions, to identify system bottlenecks, to compare design alternatives, and to determine the maximum number of clients that can be supported by the system without violating system performance requirements. Specifically, we studied the performance of the network-based distributed system for four usage patterns: NFS, Login, Rlogin and a combination of Login and NFS. The effect of the user interface (workstation, X terminal) and its configuration (memory size, disk storage) on system performance was also studied.

6 Acknowledgments
I would like to thank R. Cornejo, C. Garland, E. Bordelon, and A. Meganathan, for their help in performing the experiments described in Section 2.

References
Figure 2. *NFS workload:*
Nine users on workstations.

Figure 6. *Queueing network model of the system for diskless clients*
Figure 7. Response time (sec) as a function of the number of users in the system.

Figure 3. NFS workload: Effect of workstation configuration.

Figure 7. Response time (sec) as a function of the number of users in the system.
Figure 4. Login workload: Single user on a X terminal.

Figure 8. Server CPU utilization as a function of the number of users in the system
Figure 5. Login/NFS workload

Figure 9. NFS accelerator effect