Efficient LAN Simulation Through Node Aggregation

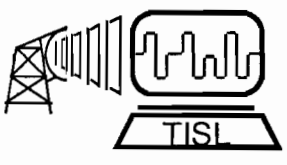
Ronald W. Cosner
Victor S. Frost

Telecommunications and Information Sciences Laboratory
University of Kansas, Center for Research
2291 Irving Hill Road, Nichols Hall
Lawrence, Kansas 66045

Technical Report

TISL TR 8000-1

April 1989



Abstract

Simulation of computer networks requires large amounts of computer resources. Modeling similar nodes in the network as a single node (supernode) has the potential to reduce the computer resources required. The purpose of this project was to determine the reduction in computer time obtained using this approach, while the results remain within the acceptable percent error. Three different Ethernet configurations were evaluated for a variety of loads and supernode combinations. The percent of CPU time saved was determined for each experiment. A method was developed for predicting the CPU time for any simulation.

Project Overview

Simulatations of large computer networks requires great amounts of computer resources. In an attempt to devise a method of reducing the CPU time required to simulate a network, this project was designed to combine like nodes of a network into a supernode with the same traffic intensity of the combined nodes. The performance of this approach was defined as the percent error determined as a function of load for each of the reduced networks. The project was divided into three separate experiments, each having a different network under investigation. In the first experiment, a 24 node homogeneous network was the target system. Thereafter, the network was reduced by combining a number of nodes into a supernode. The second experiment consisted of a 24 node heterogeneous network with three different node classes. The network reduction cases for this experiment involved three supernodes – one of each node class. The last experiment involved 40 nodes of character traffic and 6 nodes of file/page traffic for the baseline system. For the reduction cases, supernodes were created in each of the two node types. A method for predicting the time required to run a simulation was also developed. Knowing the approximate CPU time for a simulation is useful for budgeting CPU time.

2.0 Analysis of Results

This section of the report will be divided into three parts for the separate experiments of the project. For experiments one and two, the Integrated CSMA-CD model of the general local area network analysis system (GLAS) was used to run the simulations. The integrated models require both Media Access and Link Access parameters. The parameters used in the simulations are from the IEEE 802.3 standards. The values used in the above protocols are given in Table 1.

Media Access Parameter

Velocity of media as fraction of light speed.		880.0E–3
Line rate.	(Mbps)	10.00
Maximum number of collisions.	(integer)	16
Backoff limit.	(integer)	10
Interframe spacing.	(uS)	9.600
Jam time.	(uS)	3.200
Slot time.	(uS)	51.20
Maximum packet size.	(bits)	12144
Minimum packet size.	(bits)	512
Number of overhead bits per packet	(bits)	256

Link Access Parameters

Window size (sequence number range).	(integer)	8
Timeout period.	(uS)	1.000E+06
Probability of a bit being in error.	(real)	0.000
Number of network nodes.	(even integer)	24
Overhead bits per pkt, also S-frame size.	(bits)	48
Acknowledgement timeout period at receiver.	(uS)	0.000

Table 1

Load	24 Node CPU Time	Allowable Error	Acceptable Supernode Case	CPU Time Supernode Case	Percent Reduction
0.1	10.39 hrs.	3.0%	6 nodes	3.47 hrs.	68.25%
0.2	26.96 hrs.	5.0%	6 nodes	7.425 hrs.	72.46%
0.3	54.96 hrs.	5.0%	20 nodes	47.48 hrs.	13.60%

(At a load of 0.4, there were no cases that fell within the allowable error.)

Table 4

The above results show that for a homogeneous network at low loads large percentages of CPU time can be saved by using this supernode concept. As the load was increased, the error for the supernode cases increases above the acceptable error as can be seen in the plots of this network. The percent reduction of CPU time is roughly the number of nodes that were cut from the network divided by the number of nodes in the original network.

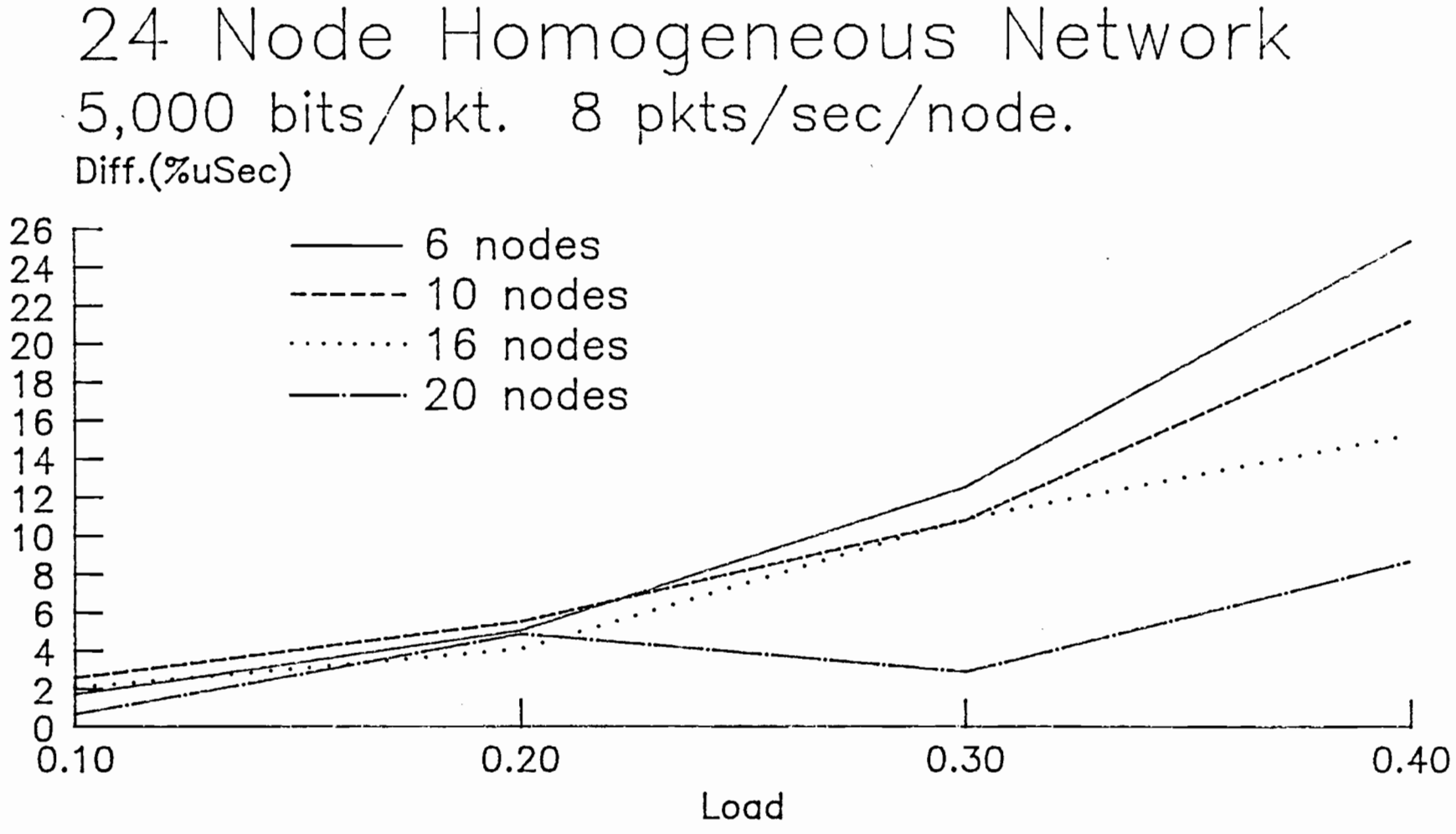

24 Node Homogeneous Network
5,000 bits/pkt. 8 pkts/sec/node.
Diff.(%uSec)
6 nodes
10 nodes
16 nodes
20 nodes
26
24
22
20
18
16
14
12
10
8
6
4
2
0
0.10
0.20
0.30
0.40
Load

Figure 1

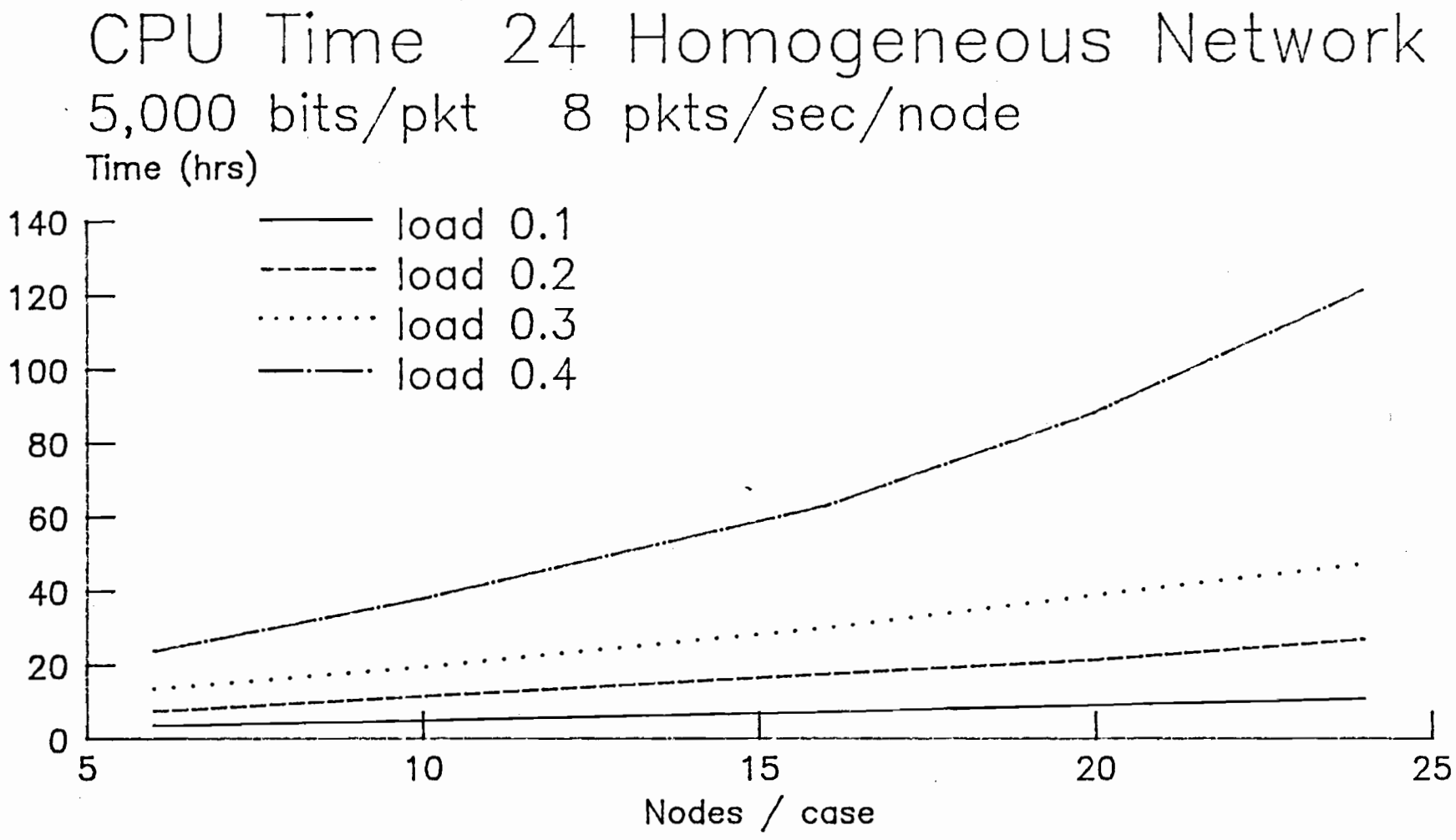

CPU Time 24 Homogeneous Network
5,000 bits/pkt 8 pkts/sec/node
Time (hrs)
load 0.1
load 0.2
load 0.3
load 0.4
140
120
100
80
60
40
20
0
5
10
15
20
25
Nodes / case

Figure 2

2.2 24 Node Heterogeneous Network

The next network configuration considered contained 24 evenly spaced nodes with three separate clusters of eight. The packet length and interarrival times are an exponential distribution. The three types of nodes on this network are described in Table 5.

Type	Packet Length	Arrival Rate
Type 1	10,000 bits/packet	4 packets/sec
Type 2	5,000 bits/packet	8 packets/sec
Type 3	2,500 bits/packet	16 packets/sec

Table 5

The baseline 24 node simulation consisted of 8 nodes of each type positioned uniformly on the network. Table 6 shows an example of a type 2 node at a load of 0.1 for the network.

Node #16

Message length distribution. (CONSTANT, EXPON)		EXPON (XMEAN)
-- Average message length	(bits)	5000
Application layer buffer size.	(messages)	20
Message destination node.	(integer)	12
Link distance.	(meters)	2.000
Bus distance to following node if appl.	(meters)	20.83
Interarrival time distribution.		EXPON (XMEAN)
-- Mean interarrival time.	(uS)	125.0E+03

Table 6

The reduced networks are formed when several nodes in a cluster are combined into one node. This is done for every cluster simultaneously. The packet length of the supernodes will remain the same, but to keep a constant load on the network, the arrival rate will be increased to account for the

merging of the combined nodes. The first case of this experiment is an 18 node network and is shown in Table 7.

Cluster	Packet Length	Arrival Rate
5 nodes	type 1	
1 node	10,000 bits/packet	12 packets/sec
5 nodes	type 2	
1 node	5,000 bits/packet	24 packets/sec
5 nodes	type 3	
1 node	2,500 bits/packet	48 packets/sec

Table 7

As can be seen from the table, the supernode is a combination of three nodes. The packet length for each cluster remains the same, the arrival rate of the supernode is increased by a factor of three to compensate for the three combined nodes. For the second case, the supernodes will be the combination of five nodes. Here are the node configurations for case two, which is a 12 node network as shown in Table 8.

Cluster	Packet Length	Arrival Rate
3 nodes	type 1	
1 node	10,000 bits/packet	20 packets/sec
3 nodes	type 2	
1 node	5,000 bits/packet	40 packets/sec
3 nodes	type 3	
1 node	2,500 bits/packet	80 packets/sec

Table 8

As in the first experiment, the acceptable error is calculated by averaging the confidence intervals of the nodes of the same cluster except for the supernode. This value is then divided by the packet length in uSec to obtain the percent. The plots were evaluated to determine which supernode cases were within the acceptable error. Each cluster was evaluated for each case and load. The plots of the percent difference of the supernode cases verses load may be viewed in Figures 3 and 4. The calculated values for the acceptable error are shown in Table 9 for each type of node and load of the network.

	Load	0.1	0.2	0.3	0.4
Allowable					
Error	10,000	3.5%	3.0%	3.0%	4.0%
per	5,000	4.0%	3.5%	4.0%	5.0%
Packet Length	2,500	4.0%	4.5%	7.0%	6.0%

Table 9

From these results and the acceptable errors, the network can be reduced to 12 nodes at loads of 0.3 or less. At a load of 0.4, neither the 12 node nor the 18 node networks are within the error margin for any of the node types. From the CPU time plot (Figure 5), the following values in Table 10 for the reduction in CPU time for the supernode networks were determined.

Load	24 Node CPU Time	Acceptable Supernode Case	CPU Time Supernode Case	Percent Reduction
0.1	12.35 hrs.	12 nodes	6.86 hrs.	44.45%
0.2	31.80 hrs.	12 nodes	17.82 hrs.	43.96%
0.3	72.50 hrs.	12 nodes	36.56 hrs.	49.57%

(At a load of 0.4, there were no cases that fell within the allowable error.)

Table 10

The above results show that for load of 0.3 or less one could save almost 50% of CPU time if the 12 node reduction network was simulated to obtain results within acceptable error of the original 24 node network. From these results, it was observed that for the smaller packets in the network the percent error was considerably greater at higher loads. One can also see that the CPU time for the different networks is very linear as the network is reduced. The changes in load only affect the slope of the plot. Again, the amount of CPU time saved is roughly the the percent of nodes cut out of the network.

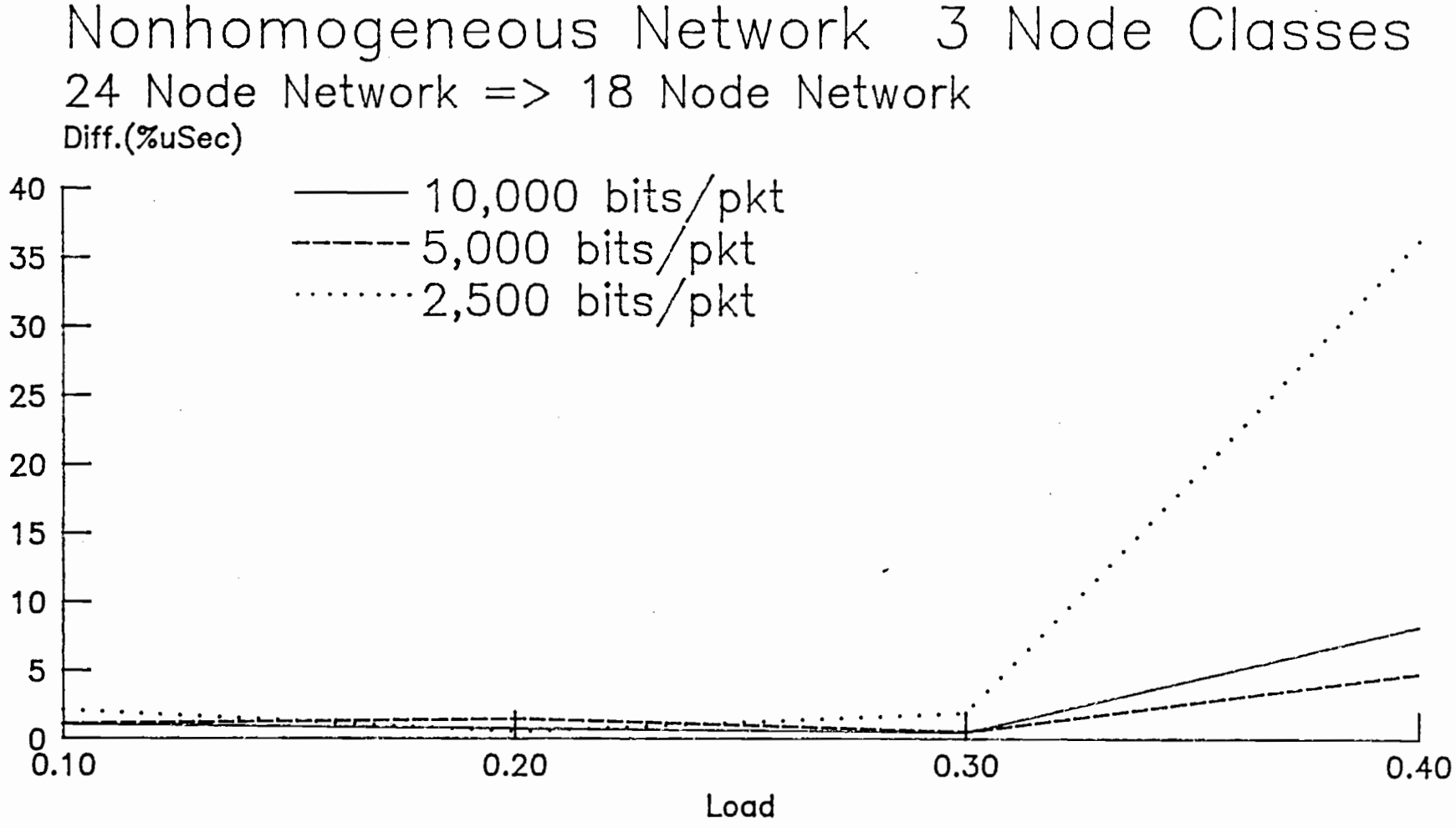

Nonhomogeneous Network 3 Node Classes
24 Node Network => 18 Node Network
Diff.(%uSec)
10,000 bits/pkt
5,000 bits/pkt
2,500 bits/pkt
40
35
30
25
20
15
10
5
0
0.10
0.20
0.30
0.40
Load

Figure 3

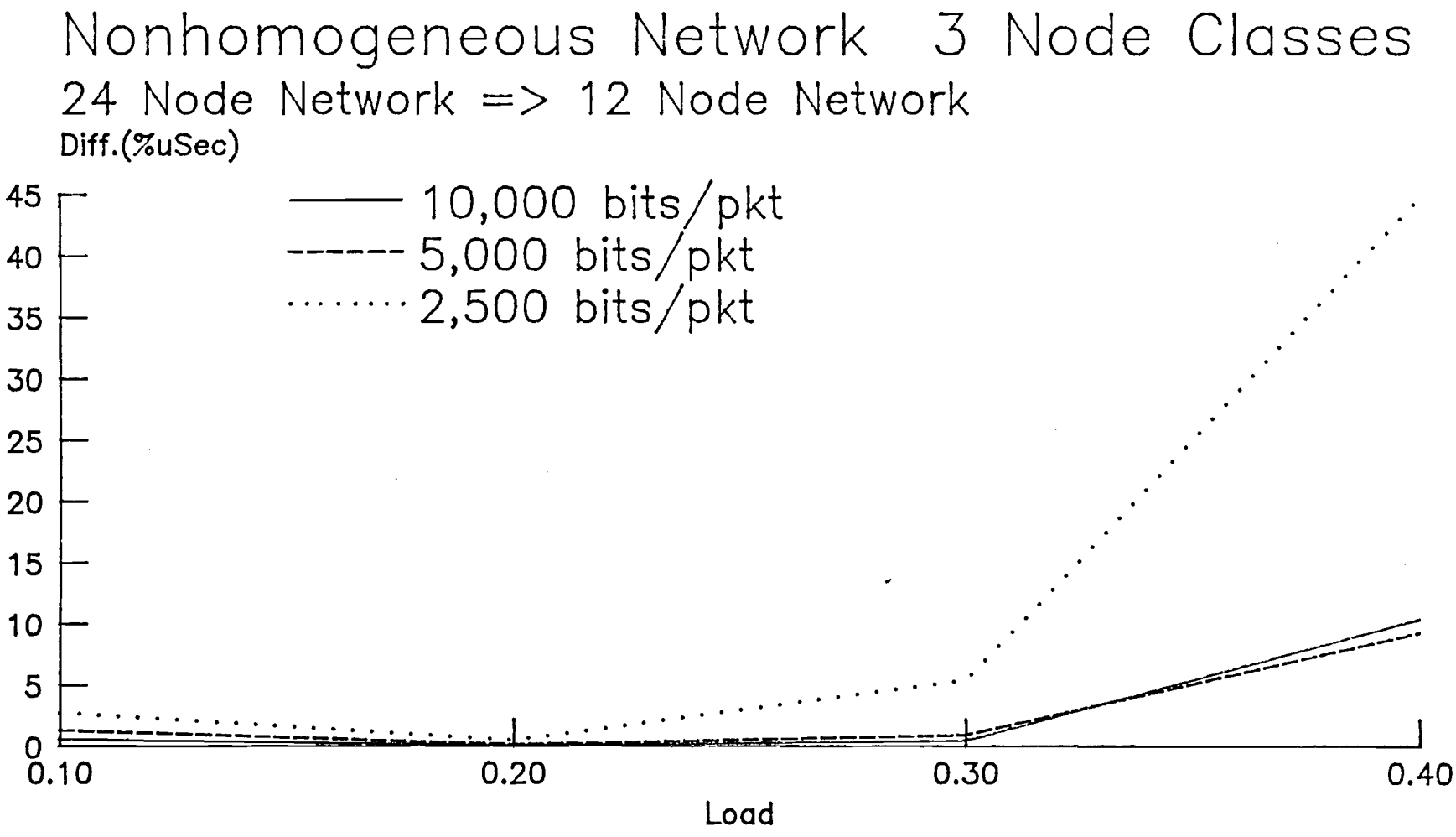

Nonhomogeneous Network 3 Node Classes
24 Node Network => 12 Node Network
Diff.(%uSec)
10,000 bits/pkt
5,000 bits/pkt
2,500 bits/pkt
45
40
35
30
25
20
15
10
5
0
0.10
0.20
0.30
0.40
Load

Figure 4

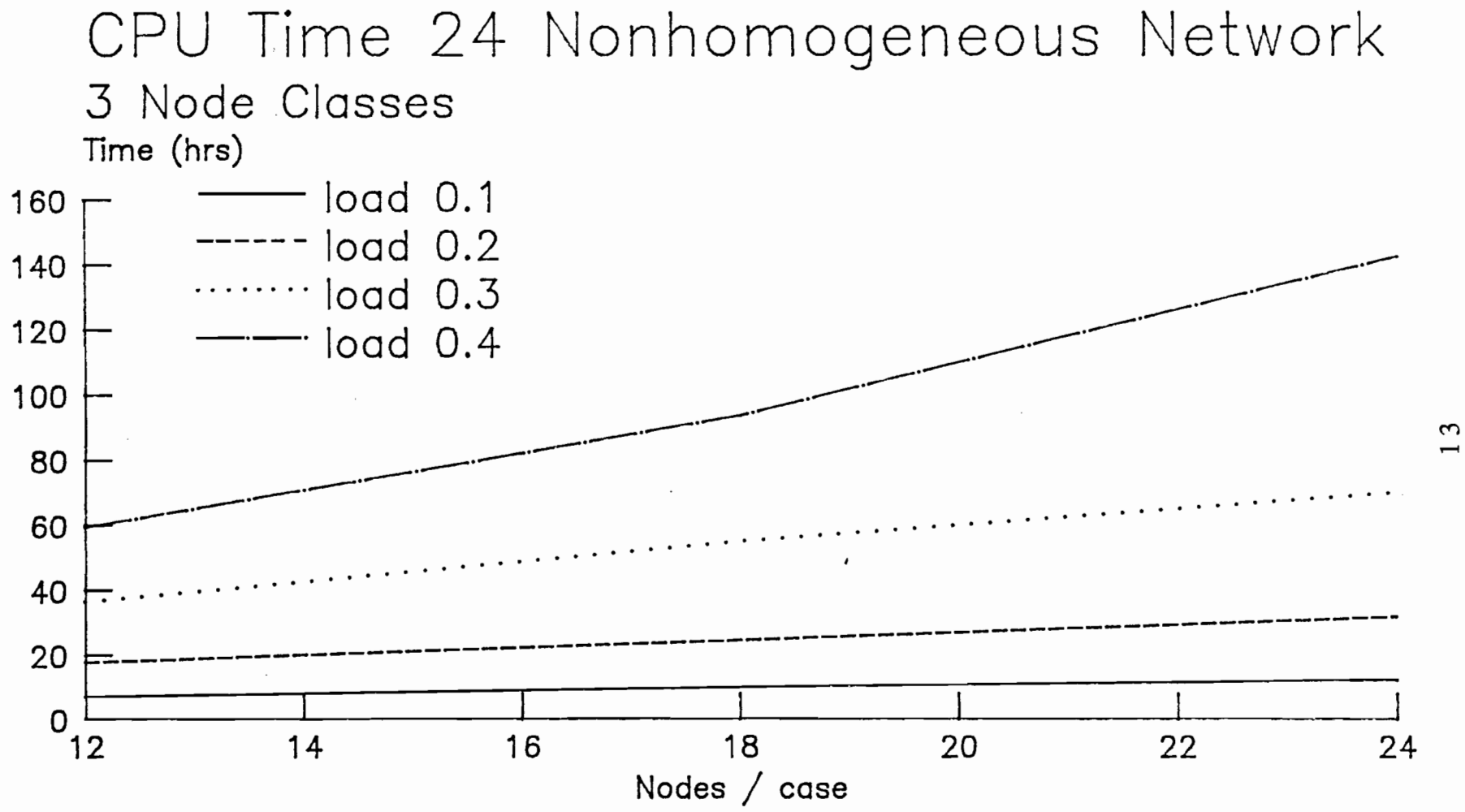

CPU Time 24 Nonhomogeneous Network
3 Node Classes
Time (hrs)
load 0.1
load 0.2
load 0.3
load 0.4
160
140
120
100
80
60
40
20
0
12
14
16
18
20
22
24
Nodes / case

Figure 5

2.3 46 Node Heterogeneous Network

The network in experiment 3 contains 46 evenly spaced nodes of two types. This part of the project is to simulate a real world network. For example, 40 nodes of type 1 would separate character traffic and 6 nodes of type 2 would separate file/page traffic. The message length and interarrival times are assured to be exponential. These two types of nodes are described in Table 11.

Type	Packet Length	Arrival Rate
Type 1	1,000 bits/packet	1.724 packets/sec
Type 2	6,000 bits/packet	25.86 packets/sec

Table 11

As can be seen from the table above, the arrival rate of the type 2 nodes is 15 times greater than that of the type one nodes. An example of a node of type one is shown in Table 12.

Node #1

Node type.	(terminal, server, host)	terminal
-- Average message length	(bits)	1000
Application layer buffer size.	(messages)	200
Link distance.	(meters)	2.000
Bus distance to following node if appl.	(meters)	10.87
Interarrival time distribution.		EXPON (XMEAN)
-- Mean interarrival time.	(uS)	580.0E+03

Table 12

The supernodes of this network were formed by combining several nodes of one group into one supernode. As in the other experiments, the packet

size remained the same, but the arrival rate was increased to keep the load constant. The configurations for this experiment are given in Table 13.

	Type 1	Type 2
Baseline System	40	6
Case 1	30	4
Case 2	20	2
Case 3	9	1

Table 13

The acceptable error was calculated in the same manner as the previous experiments. The calculated values are shown below in Table 14.

	Load	0.1	0.2	0.3	0.4
Allowable					
Error	6,000	1.5%	.83%	.83%	.83%
per	1,000	12.0%	12.0%	13.0%	12.0%
Packet Length					

Table 14

The error margin was determined from the results shown in Figures 6 and 7. It was found that the 46 node network could be reduced to a 10 node network for load of 0.2 of less. For loads of 0.3, the network could be reduced to a 34 node network. And, for loads greater than 0.3, the 46 node network could not be reduced without exceeding the error margin. From the CPU time results (Figure 8), the percent reduction of CPU time was evaluated to obtain the results shown in Table 15.

Load	24 Node CPU Time	Acceptable Supernode Case	CPU Time Supernode Case	Percent Reduction
0.1	42.54 hrs.	10 nodes	10.77 hrs.	74.69%
0.2	81.06 hrs.	10 nodes	23.06 hrs.	71.55%
0.3	139.41 hrs.	4 nodes	101.84 hrs.	26.95%

(At a load of 0.4, there were no cases that fell within the allowable error.)

Table 15

The results show that a reduction of over 70% in CPU time was obtained for loads of 0.2 or less by condensing the 46 node network into 10 nodes. For a load of 0.3, the original network was reduced in 34 nodes while the percent delay differences were still within the error margins. From these results, one can see that the percent difference for the smaller packets is again much greater than the percent delays for the larger packets at higher loads. Also, as the load was increased, the percent delays for the both packet sizes began to increases very sharply. For this experiment, the CPU time plots were linear and had increasing slope with increasing loads as in the previous experiments.

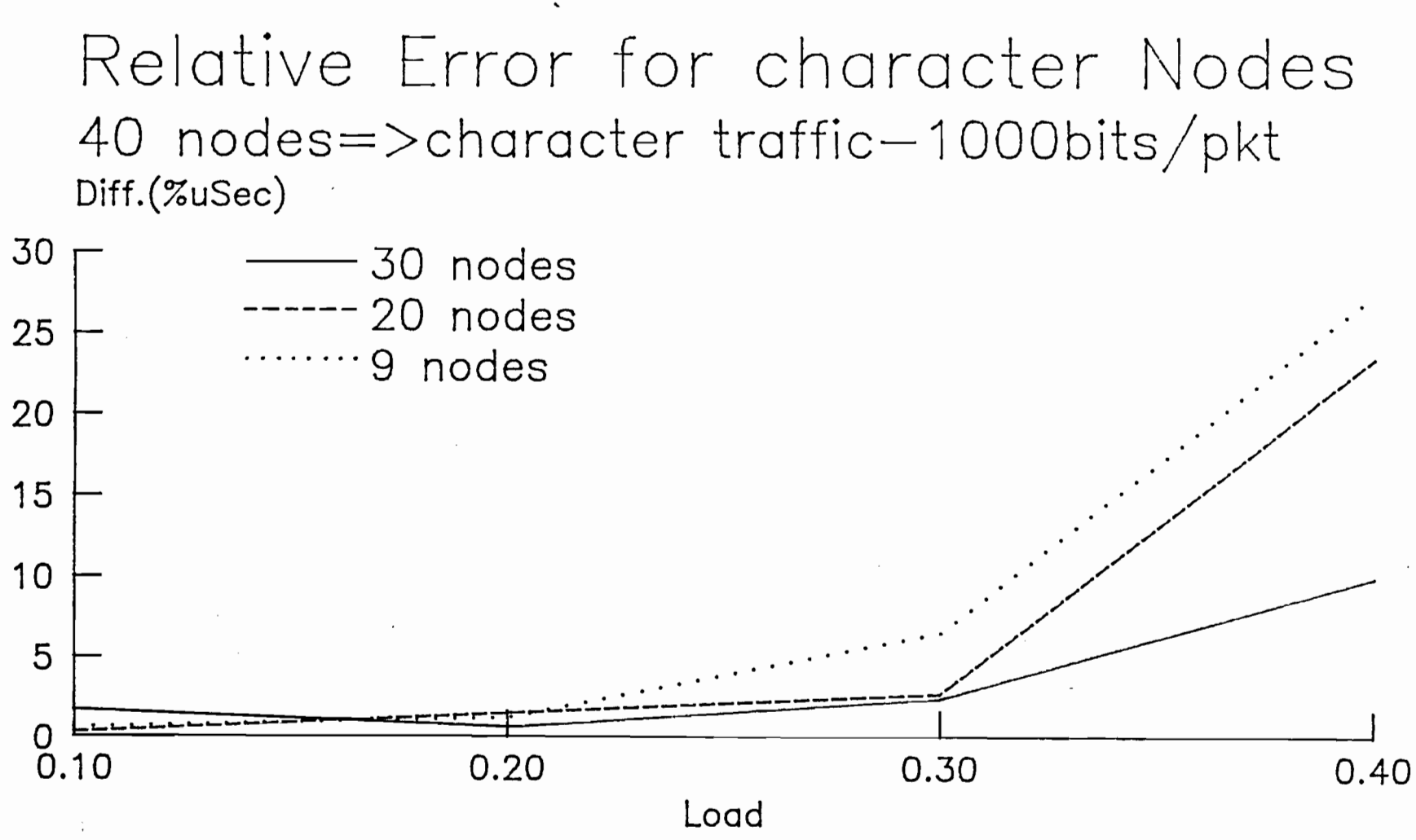

Relative Error for character Nodes
40 nodes=>character traffic—1000bits/pkt
Diff.(%uSec)
30 nodes
20 nodes
9 nodes
30
25
20
15
10
5
0
0.10
0.20
0.30
0.40
Load

Figure 6

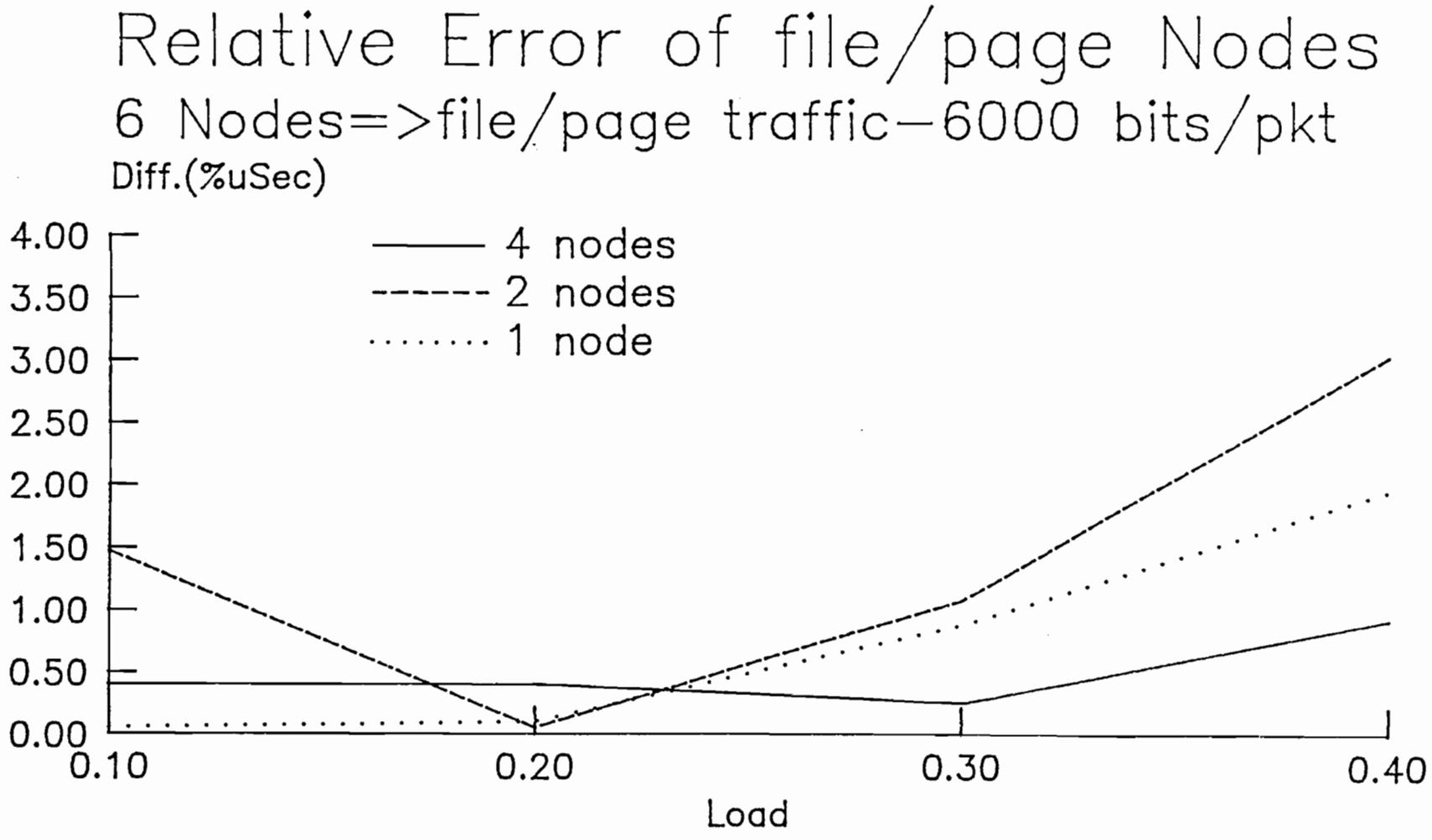

Relative Error of file/page Nodes
6 Nodes=>file/page traffic-6000 bits/pkt
Diff.(%uSec)
4 nodes
2 nodes
1 node
4.00
3.50
3.00
2.50
2.00
1.50
1.00
0.50
0.00
0.10
0.20
0.30
0.40
Load

Figure 7

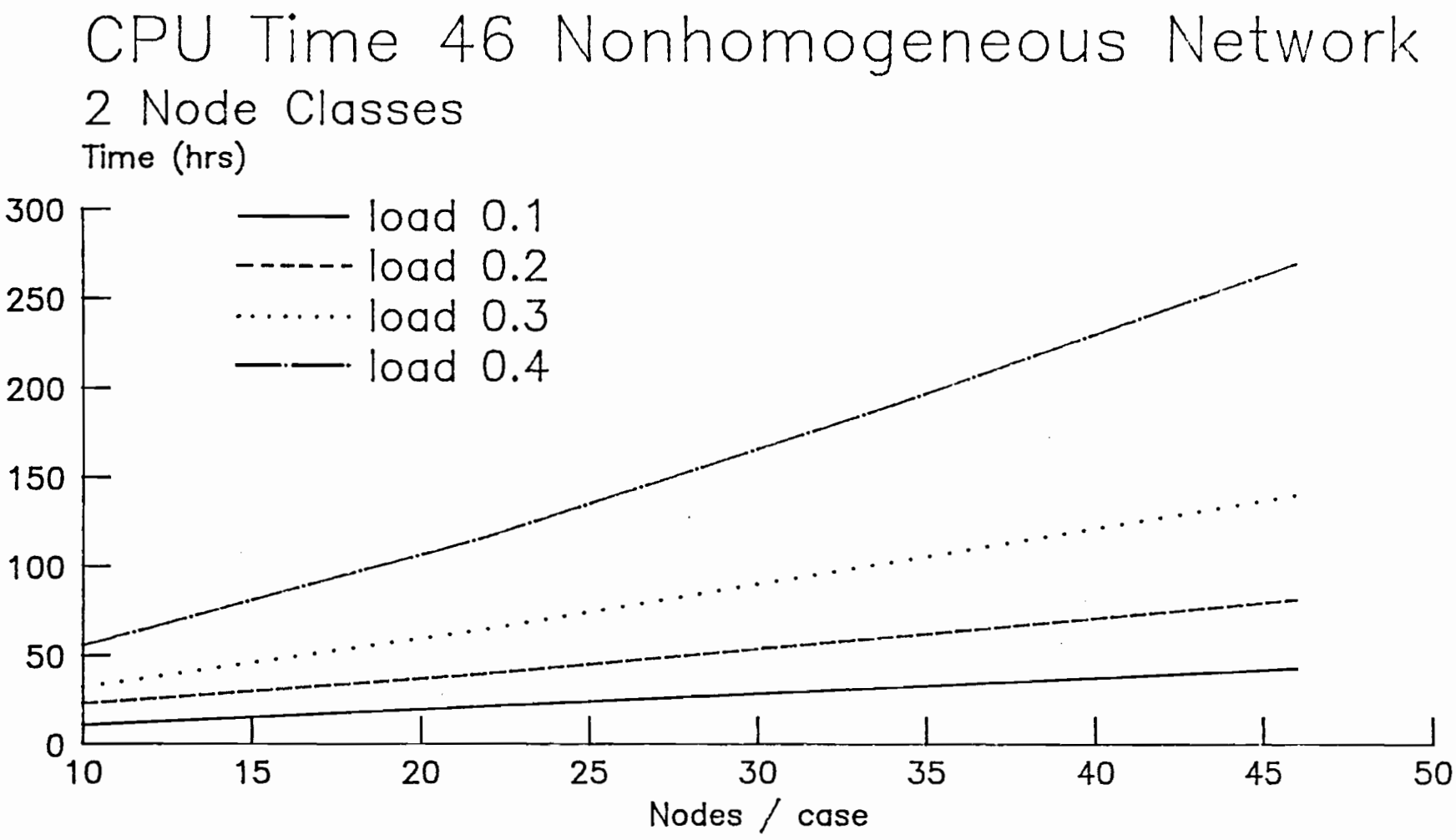

CPU Time 46 Nonhomogeneous Network
2 Node Classes
Time (hrs)
load 0.1
load 0.2
load 0.3
load 0.4
300
250
200
150
100
50
0
10
15
20
25
30
35
40
45
50
Nodes / case

Figure 8

2.4 Time Estimations

When large networks are to be simulated, it is a good idea to know approximately how much CPU time the simulation will use. This allows users of a particular computer system to budget for the long simulations. Performance statistics for these simulations were determined by running 10 subruns and generating between 100 and 200 packets per subrun per node. One way to determine the amount of CPU time required with a great deal of accuracy is to cut the number of subruns to two and the time for each subrun. By reducing the time per subrun, fewer packets will be generated. This short simulation is then run and the amount of CPU time is obtained for this condensed simulation. Any performance measures from this short simulation will be inaccurate due to the fact that only a few packets were generated. The CPU time for the short run is multiplied by five for the reduction in subruns and then multiplied by the factor that the time per subrun was reduced. This will give a very accurate predication of the total CPU hours without using large quantities of computer time.

3.0 Conclusion

In conclusion, the results of these three experiments indicate that a reduction of network size is applicable for loads of 0.3 or less. For the first experiment, reductions of approximately 70% of CPU effort was accomplished for loads of 0.1 and 0.2. Only a 13% reduction was obtained for a 0.3 load. The second experiment resulted in at least a 40% savings in CPU effort for loads of 0.3 and less. For the load of 0.4, there were no cases that fell within the allowable error margin. Experiment three showed reductions of at least 70% for load of 0.2 or less. A reduction of 27% was found for a load of 0.3 and there were no cases within the error margins for a 0.4 load.

Upon completion of this project, a very accurate method for determining the amount of CPU hours was obtained for each simulation. This is done by dividing the number of subruns and the time per subrun by some desired factors to generate only a few packets per node. These factors are then multiplied by the CPU time of the condensed simulation. This method is of great value when the CPU time is on a budget or many users have access to the computer. CPU time can then be allotted for a simulation. A simulation can also be killed if it overly exceeds its expected simulation time, meaning that something obviously is incorrect.