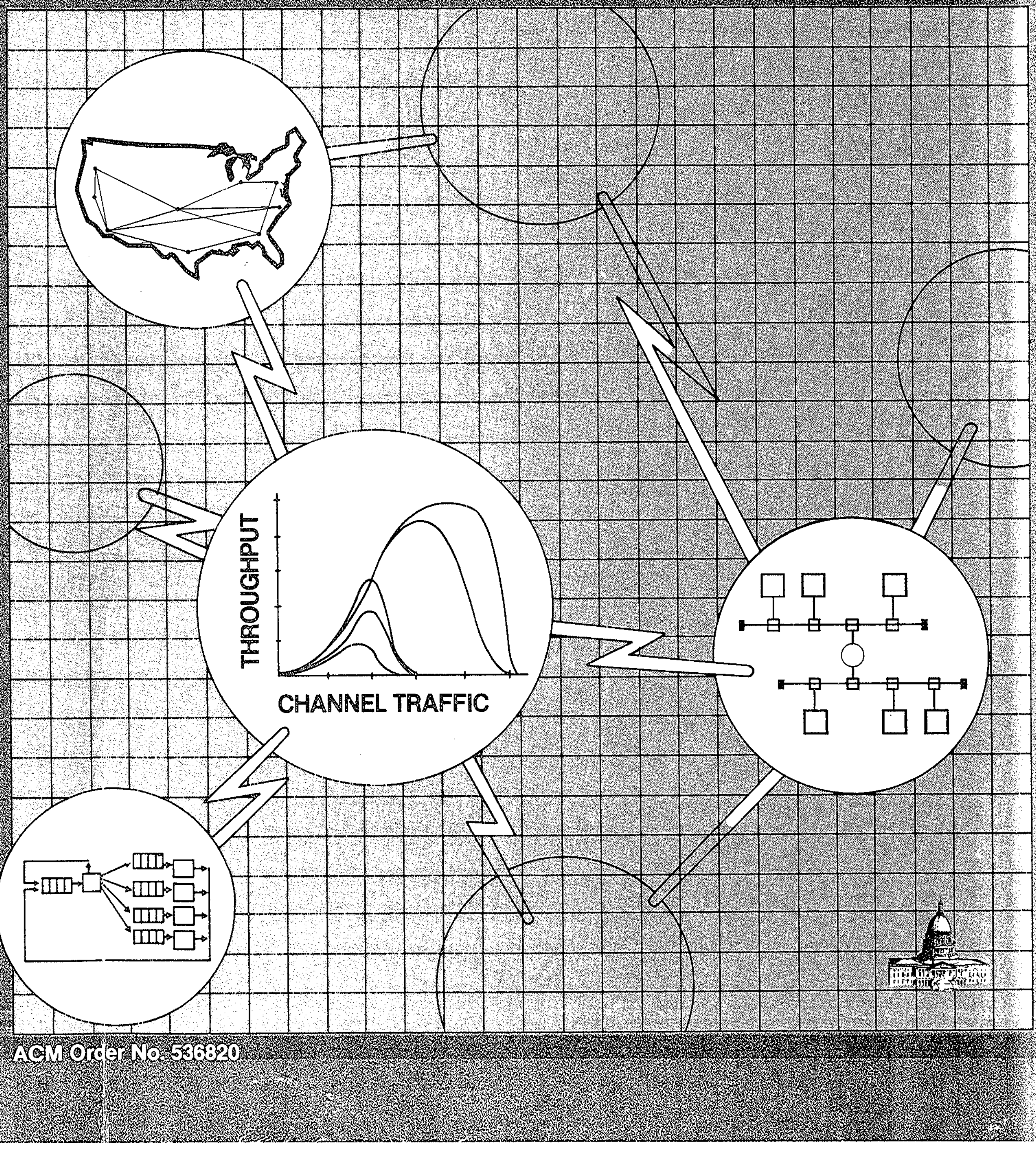
COMPUTER NETWORK PERFORMANCE SYMPOSIUM

Sponsored By SIGCOMM, SIGMETRICS, & SIGOPS And The University of Maryland
College Park, Maryland April 13-14, 1982

Performance Evaluation Review, Volume 11, Number 1 Spring 1982



ACM Order No. 536820

COMPUTER NETWORK PERFORMANCE SYMPOSIUM

Sponsored By SIGCOMM, SIGMETRICS & SIGOPS And The University Of Maryland

College Park, Maryland
April 13-14, 1982

A Publication of the ACM
Special Interest Group on
Measurement and Evaluation

Volume 11 Number 1
Spring 1982

PERFORMANCE EVALUATION REVIEW

ACM Order No. 536820

The Association for Computing Machinery
11 West 42nd Street
New York, New York 10036

Additional copies of these Proceedings may be ordered prepaid from:

ACM Order Department
P.O. Box 64145
Baltimore, MD 21264

Please state the ACM Order Number:

Prices:	ACM Members:	$12.00 prepaid
	All others:	$16.00 prepaid

ACM Order No. 536820

ISBN 0-89791-069-9

Load Balancing
in
Homogeneous Broadcast Distributed Systems

by
Miron Livny and Myron Melman
Department of Applied Mathematics
The Weizmann Institute of Science
Rehovot, Israel

ABSTRACT

Three different load balancing algorithms for distributed systems that consist of a number of identical processors and a CSMA communication system are presented in this paper. Some of the properties of a multi-resource system and the balancing process are demonstrated by an analytic model. Simulation is used as a mean for studying the interdependency between the parameters of the distributed system and the behaviour of the balancing algorithm. The results of this study shed light on the characteristics of the load balancing process.

INTRODUCTION

Distributed processing systems are characterized by resource multiplicity and system transparency [1]. Every distributed system consists of a number of autonomous resources that interact through a communication system. From the user's point of view this set of resources acts like a 'single virtual system'. As he submits a task for execution he does not and should not consider either the internal structure or the instantaneous load of the system. It is the duty of the system's load balancing algorithm to control the assignment of resources to tasks and to route the tasks according to these assignments.

The stochastic properties of the tasks - arrival and execution times - cause resource contentions that lead to the establishment of queues. The existence of queues of waiting tasks demands dynamic reconsideration of previous assignments. The assignment algorithm is motivated by the desire to achieve better overall performance relative to some selected metric of system performance. The strategy of the load balancing algorithm has a strong effect on the utilization of the system resources and determines its overall performance. The purpose of this paper is to investigate the behaviour of the load balancing process in broadcast distributed systems.

The problem of resource allocation in an environment of cooperating autonomous resources and its relationship to system performance is a major issue associated with the design of distributed systems [2]. A number of studies of this issue have been reported [3] [4] [5] [6]. Most of these studies deal with distributed systems that utilize central elements, such as a job dispatcher, a shared memory , a main processor, or with systems that consist only of two processors. This paper deals with distributed load balancing algorithms for homogeneous distributed systems whose communication system consists of a broadcast medium. There are no central elements in the system and the balancing algorithm is distributed among the resources. The policy of the algorithm is to minimize the expected turnaround time of the tasks.

Initially a simple analytic model is used for demonstrating some of the properties of a multi-resource system and the balancing process. Then three different load balancing algorithms for broadcast distributed systems are defined and discussed. The last part of the paper presents results of the simulation study. In the study, the three algorithms were simulated under various operating conditions. The results demonstrate the interdependency between the parameters of the distributed system and the behaviour of the balancing algorithm.

© 1982 ACM 0-89791-069-9/82/0400-0047 $00.75

LOAD BALANCING

In a distributed system it might happen that a task waits for service at the queue of one resource while at the same time another resource which is capable of serving the task is idle. A load balancing algorithm whose goal is to minimize the expected turnaround time of the tasks will tend to prevent the system from reaching such a state.

Assume a system of N identicall and independent M/M/1 queueing systems [7]. Let P_{wi} be the probability that the system is in a state in which at least one customer waits for service and at least one server is idle then

$$P_{wi} = \sum_{i=1}^{N} \binom{N}{i} Q_i H_{N-i} = (1-Po^N)(1-Po^N-(1-Po)^N)$$

where

$Q_i = Po^i$ is the probability that a given set of i servers are idle

$H_i = (1-Po^i)(Po(1-Po))^i$ is the probability that a given set of i servers is not idle and at one or more of them a task waits for service

$Po = 1-\frac{\lambda}{\mu}$ is the probability that a server is idle.

Fig. 1 shows the value of P_{wi} for various values of server utilizations, $\rho = 1-Po$, and number of servers N. The curves of the figure indicate that for practical values of ρ, P_{wi} is remarkably high and that in systems with more than ten servers almost all the time a customer is waiting for service and another server is idling.

The high value of P_{wi} indicates that by balancing the instantaneous load of the multi-resource system their performance can be considerably improved. Note that the average load of a server is the same for all servers. The shape of the curves shows that for a given number of servers P_{wi} reaches its maximum value when the servers are utilized during 65% of the time. As the utilization of the servers increases past the level of 65% P_{wi} decreases. This property of P_{wi} indicates that a *'good'* load balancing algorithm should work less when the system is heavily utilized. It is clear that the same thing is true for systems that are idle most of the time.

A reduction in P_{wi} of a multi-resource system will cause an improvement *of* the expected turnaround time, W, of the tasks. If the servers are interconnected by a communication system P_{wi} can be

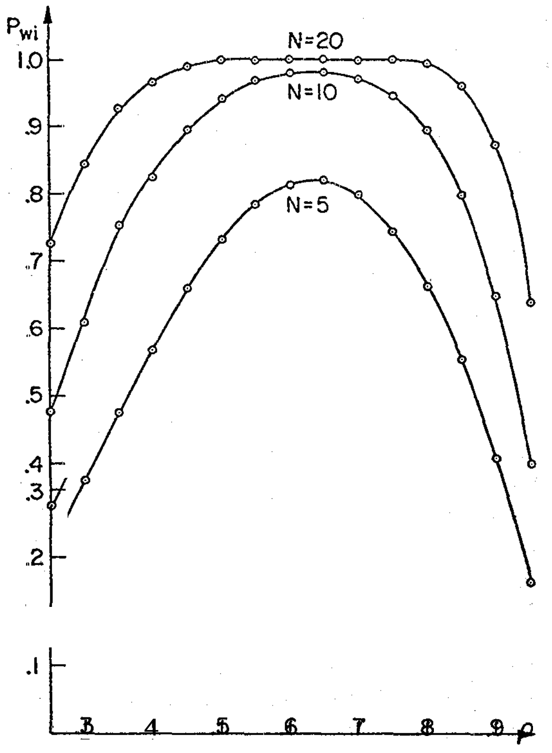


figure: 1 P_{wi} as a function of ρ

reduced by transfering tasks from one queue to another. These transfers affect the utilization and consequently the performance of the communication system and can be considered as the price paid for the reduction of W.

The expected turnaround time of the above multi-resource system will be minimal if P_{wi} will be zero. In such a case the system will behave like an M/M/N (single queue N servers) system [7]. P_{wi} can be reduced ***to*** zero only if the servers are inter-connected by a communication system whose task transfer rate is much higher than the service rate of the servers. In a system where P_{wi} is zero a task will be transferred from one queue to another when one of the following events occurs:

1. A task arrives at a busy server and there are less than N tasks in the system.
2. A server completes the service of a task, no other tasks are waiting in its queue and there are more than N tasks in the entire system.

Therefore a lower bound to the rate of **tasks** transferred in order to minimize W is given by

$$LT = \sum_{i=1}^{N-1} (\lambda i P_i + \mu(N-i)P_{N+i})$$

[1] All the systems have the same arrival, λ, and service, μ, rates.

where P_i is the probability of having i tasks in an M/M/N system [7]. The first element of the summation is the rate of transfers caused by the arriving tasks (the first event). The second element is part of the transfer rate caused by the departing tasks (the second event).

Fig. 2 gives the values of the lower bound LT as a function of the number of servers for various arrival rates, λ. Note that a considerable number of tasks has to be transferred in order to achieve the performance of a M/M/N system. For systems with more than ten servers almost one out of λ^{-1} tasks are transferred.

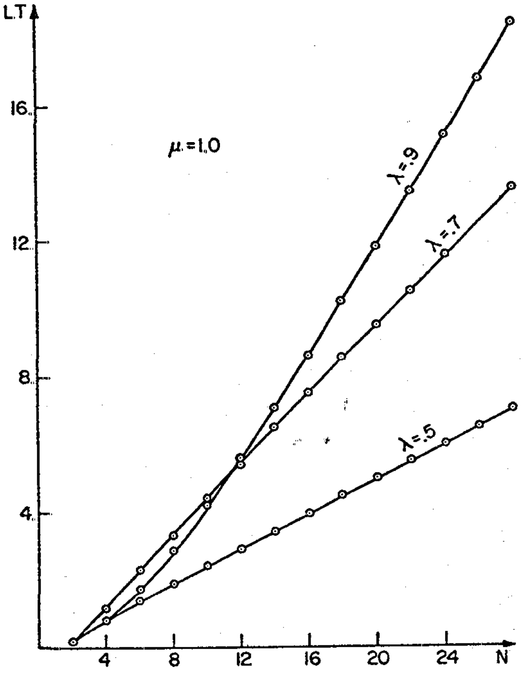


figure: 2 Lower Bound on task transfer rate in an M/M/N like system vs. nuber of servers

These results indicate that in systems where task transmission time is not negligable the load balancing process will utilize a large portion of the capacity of the communication system. The utilization of the communication system will determine the delays associated with the transmission of a task or any other message. These delays will cause an increase in P_{wi} and therefore an increase in W. The amount of traffic generated by the balancing algorithm has a major effect on its ability to improve the performance of the system. Fig. 2 shows that in order to achieve the optimal performance, $P_{wi} = 0$, a large portion of the tasks have to be transferred.

THE DISTRIBUTED SYSTEM MODEL

The model describes a homogeneous N-server distributed system. The system consists of N identical nodes and a communication channel. Every node has a processor P, a communication processor CP and a queue, Fig. 3. The channel is a passive broadcast medium (radio or coxial/fiber cable) with a CSMA-CD (carrier sense multiple access collision detection) access method. The access to the channel and the transmission of messages is controlled by the CP according to the ETHERNET protocol [8] [9].

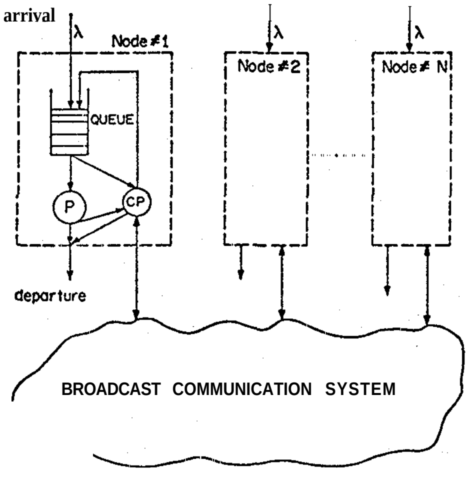


figure: 3 The Broadcast Distributed System

Tasks arrive independently at each node and join the queue. The queueing discipline at all the nodes is FIFO (first-in-first-out). The arrival rate of each stream of tasks is λ and the inter-arrival time has a negative exponential distribution. The task arrival process to the entire system consists of N identical independent poisson processes with a total rate of N .

The service time demand of the tasks has a negative exponential distribution and the mean service time is μ^{-1}. The tasks leave the system after being served, and depart from the same node at which they had entered the system. It is assumed that the system operates in steady-state conditions ($\lambda < \mu$). The utilization of the servers is $\rho = \frac{\lambda}{\mu}$.

The number of tasks at node i (waiting for service or being served) is denoted by N_i and $ST=(n_1,\ldots\ldots,n_N)$ describes the state of the system. A state of the system is defined as unbalanced if there are two servers i and j such that $n_i - n_j > 1$. The unbalance factor of a state ST is defined as

$$UBF = \begin{cases} \underset{\emptyset<i,j\leq N}{\mathrm{MAXIMUM}}((n_i-n_j)n_j^{-1}) & \text{if ST is UNB} \\ \emptyset & \text{otherwise} \end{cases}$$

Note that if the system is in an unbalanced state and one of the servers is idle the UBF of the state is infinite.

The purpose of the channel, Fig. 3, is to transfer tasks from one node to the other in order to improve the expected turnaround time of a task. The flow of tasks via the channel is governed by a distributed load balancing algorithm.

A node that wants to transfer one of its waiting tasks to another node will send it a message that describes the task. The message has to contain all the external data a server needs in order to identify and serve the task. In this model it is assumed that this amount of data, T, is fixed and the same amount of data is sent from the node that executed the task back to the entrance node of the task. Such a transmission takes place only when the task was not served by the node at which it entered the system. The balancing rate of the system, β, is defined as $\frac{\mu^{-1}}{CT}$ where C is the capacity of the communication channel. The factor β expresses the ratio between the mean execution time of a task and the time needed to transfer a task from one node to another. Note that when β is zero the system becomes an N(M/M/1) queueing system and when β becomes very large the system behaves like an M/M/N system.

LOAD BALANCING ALGORITHMS

A distributed load balancing algorithm is composed of two main elements - the control law element and the information policy element. The control law determines when, from where and to whom to transfer a waiting task. The decision is made according to the current available information on the state of the system. It is the function of the information policy to collect data for the control element concerning the load of the system resources. Both elements use the communication system for carrying out their functions. The control element sends messages that describe tasks and the information element sends 'status messages' that contain data on the system's load.

The delays associated with the transmission of a message may lead to the execution of a wrong operation by the balancing algorithm. As a result of such an operation a task is placed in a queue that has more waiting tasks than the queue from which the task has been removed. The balancing process faces a 'transmission dilemma' because of the two opposing impacts the transmission of a message has on the overall performance of the system. On the one hand the transmission improves the ability of the algorithm to balance the load. On the other hand it raises the expected queueing time of messages because of the increase in the utilization of the channel. The net impact of a message transmission on the overall performance of the system depends on the balancing rate of the communication system, the number of nodes and the rate at which tasks arrive at the system.

Three different distributed load balancing algorithms for broadcast distributed system are defined in this study. From the load balancing point of view broadcast communication systems have two advantages:

1. Uniform distance - the expected time that is needed to transfer a message from one node to another is the same for all pairs of nodes. Therefore all the nodes are equal-priority candidates for receiving a waiting task. Only the relative load of the nodes has to be considered by the control law.

2. Messages broadcast - the capability of the communication system to broadcast messages improves the ability of the algorithm to get a global and updated description of the system status.

The communication system consists of a single transmission resource and therefore it can not transfer a number of messages simultaneously. The high rate of message transfers generated by the balancing process (fig. 2) requires that the balancing rate of the system will be high.

The state broadcast algorithm - STB. The STB balancing algorithm utilizes both the broadcast and the uniform distance properties of the communication system. The information policy of the algorithm is based on status broadcast messages. Whenever the state of the node changes, because of the arrival or departure of a task, the node broadcasts a status message that describes its new state. This information policy enables each node to hold its own updated copy of the system state vector, SSV, and guarantees that all the copies are identical. The information contained in the $SSV=(s_1, .., s_N)$ gives the node a global and updated picture of the system state and enables the control law to base its decisions on the state of the whole system. Note that SSV may differ from ST due to transmission delays. The distributed control law of the STB algorithm will transfer a waiting task from node i to node j if the following conditions are fulfilled.

1 $s_i - s_j > 1 + (BT \cdot s_j)$ where BT is a parameter that controls the balancing threshold of the algorithm.

2. $((s_i > s_k)$ or $(s_i = s_k$ and $i \geq k))$ for all $k = 1,...,N$.
3. $s_j \leq s_k$ for $k = 1,....,N$.

When more than one node has a minimal number of waiting tasks the selection of the destination node is made randomly.

The broadcast idle algorithm - BID . The BID algorithm is based on a less liberal information policy. Under this policy a node broadcasts a status message when it enters an idle state. The message alerts all the other nodes and causes them to activate the control element of the algorithm. The control law of the BID algorithm consists of the following steps:

1. If $n_i > 1$ go to step 2, else terminate the algorithm.
2. Wait $D \cdot n_i^{-1}$ units of time. D is a parameter of the algorithm. Its value depends on the properties of the communication system.
3. Broadcast a reservation message if no other node has broadcasted such a message during the time-out period.[2] If another node has succeeded to broadcast a reservation message terminate the algorithm.
4. Wait for a reply message. The reply will be positive if the node that has broadcasted the idle message is still idle. The node will send a reply in any case.
5. If the reply is positive and $n_i > 1$ transfer a task to the idle node, else terminate the algorithm.

The purpose of the state-dependent time-out period is to give nodes with greater load a better chance to transfer a task to the idle node.

The poll when idle algorithm - PID. The information policy of both previous algorithms is based on broadcast messages. The information policy of the PID algorithm is based on polling . The node starts to poll a subset of the system nodes whenever it enters an idle state. The sequence of the polling operation of the PID algorithm is the following:

1. Randomly select a set of R nodes $(a_i, \ldots, a_R)$ and set $J = 1$. R is a parameter of the algorithm.
2. Send a message to node a_j and wait for a reply.
3. Receive the reply message. Node a_j will either send back one of its waiting tasks, if there are any, or an 'empty queue' reply.
4. If the node is still idle and $j < R$, increment j and go to step 2 else terminate the polling.

The STB algorithm attempts to prevent the system from being in a state in which the UBF is greater than BT whereas the two other algorithms decide to transfer a task only when the UBF of the state is infinite. The STB algorithm is motivated by the assumption that by keeping the UBF of the system below BT the probability that the system will be in a state with an infinite UBF will decrease. The IDB and PID algorithms assume that because of the 'transmission dilemma' it is more important to minimize the channel utilization than to keep the UBF below a finite level.

SIMULATION STUDY

All the above algorithms aspire to improve the performance of the distributed system by balancing the instantaneous load of the system resources, each one in its own way. In order to evaluate the algorithms their performance has to be predicted and the relation between their behaviour and the parameters of the system studied.

The balanced distributed system can be modeled as a queueing network. Because of the dynamic routing of the tasks the queueing model has no feasible numerical solution. Therefore simulation has to be used as a means to predict the performance of the model.

For this study three discrete time simulation models were written using SIMSCRIPT 11.5. Each model describes a different algorithm. In all the models it was assumed that there are no delays associated with the control operations of the balancing algorithm. The only delays considered are communication delays. The communication is carried out according to the ETHERNET protocol and the effect of collisions is included in the simulation model. Table 1 lists the numerical values of the simulation parameters.

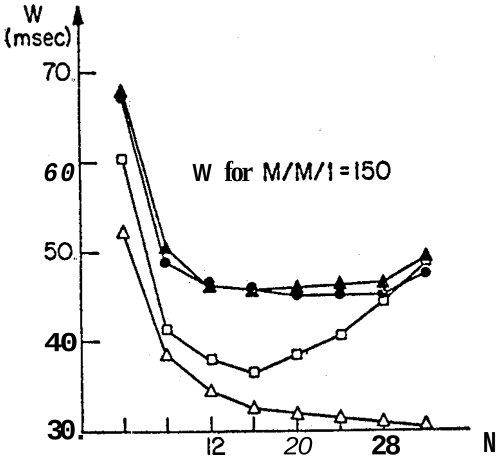
The expected turnaround time, W, of a task in an M/M/N queueing system with a task arrival rate of $N \cdot \lambda$ is a monotonic decreasing function of N [7]. Although the addition of another server increases the rate at which tasks arrive at the system the supplemental node decreases the expected queueing time of a task.

The effect of the number of nodes, N, on the W of the distributed system is demonstrated by Fig.

[2] If the transmission of the message is delayed because of collisions the same condition is tested before an attempt to retransmit the message is made.

STB - □ ; IDB- ▲ ; PID - ● ; M/M/N - △ ; M/M/1 - ○ ;

ρ=.8
β=40.



tigure: 4 expected turnaround time vs. number of servers β=4Ø

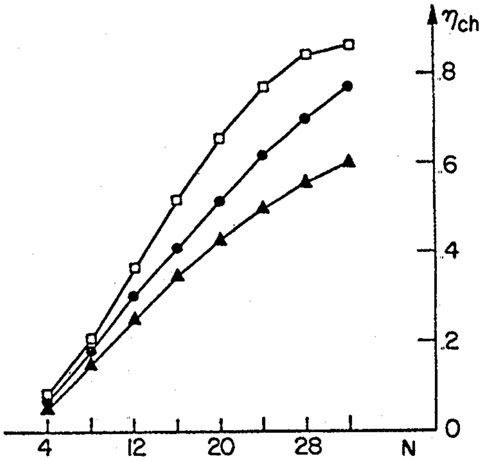


figure:4.a channel utilization vs. number of servers β =4Ø

ρ =.a
β=20.

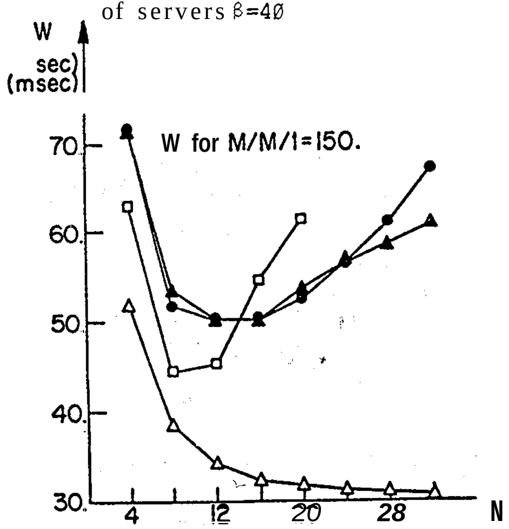
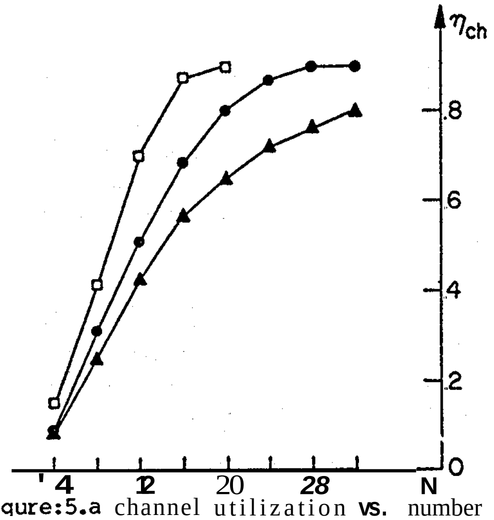


figure: 5 expected turnaround time vs. nunber of servers β=2Ø



fiqure:5.a channel utilization vs. number of servers β =2Ø

ρ=.8
β=10.

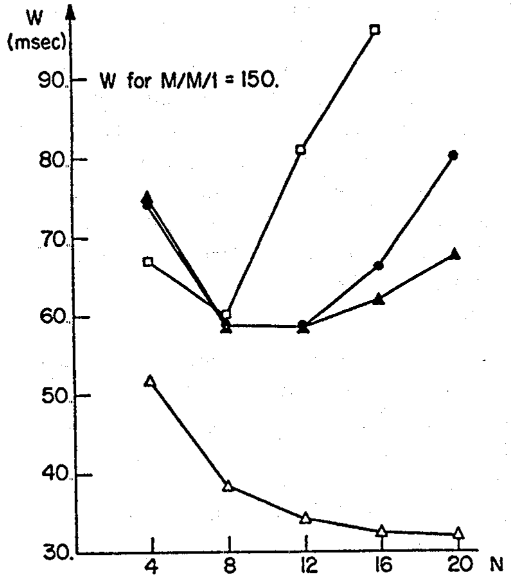


figure: 6 expected turnaround time vs. number of servers β =1Ø

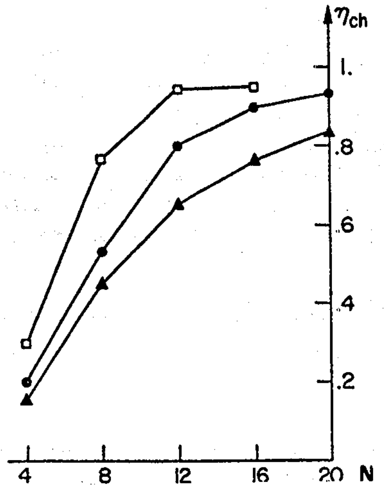


figure:6.a channel utilization vs. number of servers β =1Ø

STB-• ; IDB- A ; PID-● ; M/M/N-△ ; M/M/1-○ ;

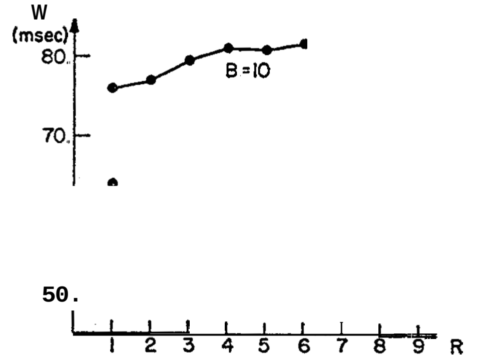


figure: 7 expected turnaround time vs. the R parameter

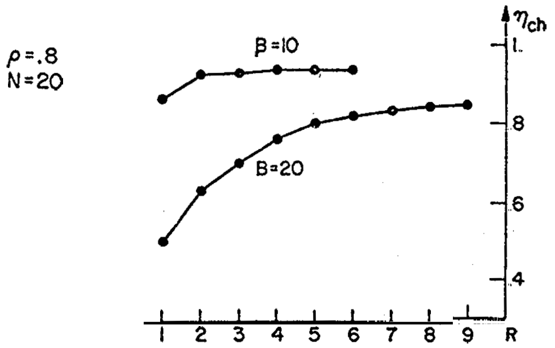


figure:7.a channel utilization vs. the R parameter

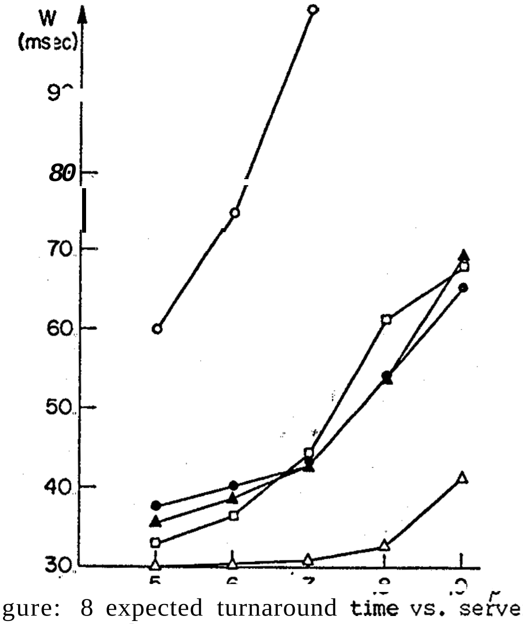


figure: 8 expected turnaround time vs. servers utilization

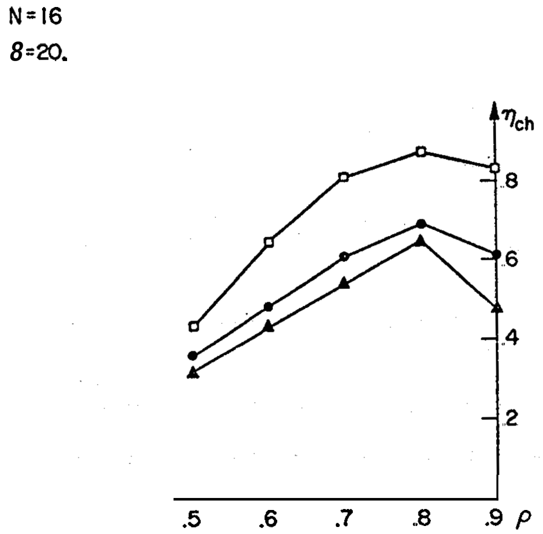


figure:8.a channel utilization VS. servers utilization

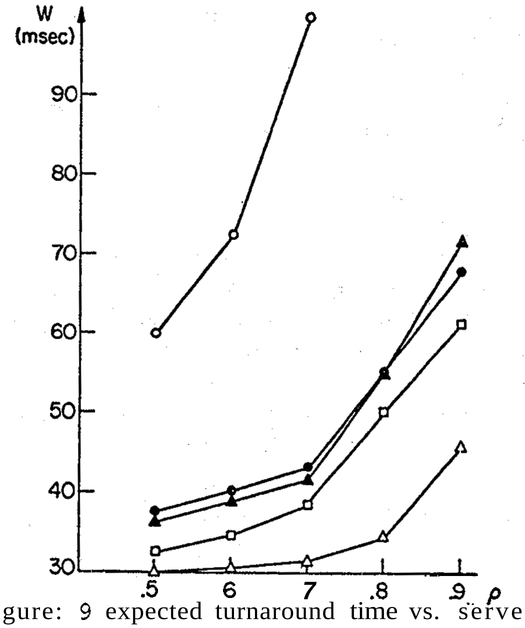


figure: 9 expected turnaround time vs. servers utilization

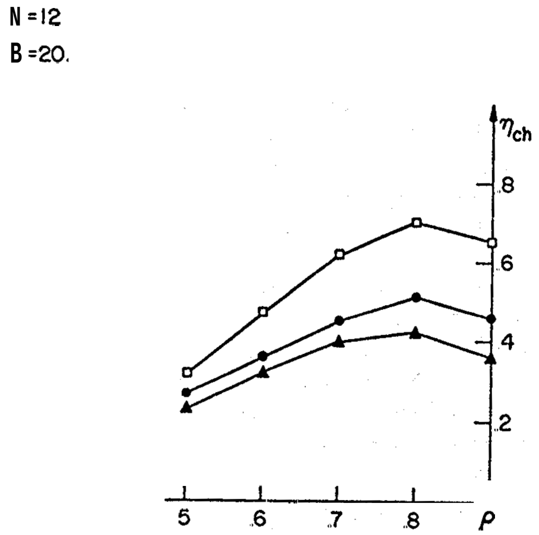


figure:9.a channel utilization vs. servers utilization

TABLE 1
Values of simulation parameters

channel transmission rate	3 Mbit/sec
slot length (see [8])	3.2 μsec
retransmission delay uniformly distributed between 28. CN sec and 50. CN sec where CN is the collision counter (see [9])	
transmission time of status/ reservation/polling message	50 μsec
expected task service time (μ^{-1})	30 msec
BT parameter of STB algorithm	1.9
D factor of IDB algorithm	1.0 msec
R parameter of PID algorithm	5
balancing rate β	1Ø,2Ø,4Ø (β=1Ø means T=1Kbyte)
simulation length. λ^{-1}	30.0 sec

4, 5, 6. The figures give the W of the three algorithms for three different balancing rates, B. In all the cases the balanced system has a considerably better W then the unbalanced system, M/M/1.

For a system with β=1Ø the expected waiting time of a task is decreased by at least 70%. The degree to which the balancing algorithm approaches the optimal W of an N server system (M/M/N) depends both on the balancing rate of the system and on the number of nodes. The turnaround time curves show that an increase in the number of nodes in a balanced distributed system has two counteracting effects. On the one hand it improves the probability that a waiting task will be transferred to an idle server, as in an M/M/N system. Rut on the other hand it raises the utilization of the communication channel, Fig. 4a, 5a, 6a. Higher channel utilization causes a slowdown in the balancing process resulting from an increase in message queueing delays. The net result of these two effects will determine whether the increase in N improves, does not affect, or deteriorates the expected turnaround time of a task. Every algorithm reaches a point, N_m, at which an addition of anothdr server will cause an increase in W. The value of N_m depends on the algorithm and balancing rate of the system. Note that in all cases when N is less than the N_m of the STB algorithm the W of this algorithm is the smallest. After it reaches its minimal value the W of the STB algorithm increases in a steep slope until it becomes greater than the W of the other algorithms. The degradation in the performance of the STB algorithm is caused by the increase in transmission delays. The BID and STP algorithms are less sensitive to the utilization of the channel. Therefore there is a wide range of N values for which they have almost the same performance. The reservation mechanism of these algorithms helps them to prevent 'wrong operations'. On the other hand the two algorithms transfer tasks only when at least one of the servers is idle. Therfore an increase in transmission delays increases the P_{wi} of the system. The IDB and PID algorithms have almost the same W under all the conditions simulated.

The balancing process utilizes a large portion of the communication channel capacity, Fig. 4a, 5a, 6a. The STB algorithm has the highest channel utilization and the IDB the smallest. The communication activity of the PLI algorithm can be easily controlled by the value of the R parameter. Fig. 7, 7a show how both channel and W depend on the size of the polling set of the algorithm. Note that for β = 10 a decrease in R causes a reduction in both W and the channel utilization.

Fig. 8 and 9 show how the balancing process reacts to changes in the utilization of the servers, ρ. For all values of ρ that wee simulated the balanced algorithms improve considerably the expected turnaround time of the tasks. Note that the relative performance of the algorithms depend on the utilization of the servers.

Fig. 8a, 9a show that when the system is heavily utilized ,ρ >.8, an increase in the utilization of the system ·causes a decrease in the channel utilization. Although the throughput of the system increases, the amount of transfers needed to balance the system decreases.

CONCLUDING REMARKS

In the opening analysis it was shown that the expected queueing time of a task in a distributed system can be reduced by means of load balancing. The results obtained from the simulation studies give a quantitative description to this ability. The results presented demonstrate the strong dependency between the performance of the balancing algorithm and the system parameters.

The purpose of the study was to shed light on the load balancing process in homogeneous broadcast distributed systems. The three algorithms that were defined in the course of the study represent three different approaches to the distributed load balancing problem. The simulation results show that each approach is the 'best' under certain conditions. The dependency between the behaviour of the algorithm and the parameters of the system deters from any attempt to select the ultimately 'best' algorithm. For these algorithms, as for other distributed control algorithms, there is no absolute answer to the question 'is algorithm A better then B' (see [1Ø]). Therefore getting a better uderstanding of the processes involved in distributed load balancing has to be the aim of a study of this type of algorithms.

Three main conclusions can be derived from the simulation study:

1. Higher resource multiplicity does not necessarily result in better turnaround time. Every algorithm reaches a point at which an increase in the number of servers decreases the performance of the system. Therefore when a number of servers is given it might be better, from the W point of view, to assemble them into two or more systems than to integrate them into one system.
2. The balancing process has a high communication activity. This has been predicted by the analytic analysis and is demonstrated by the results of the simulation runs.
3. The selection of the control law and information policy should depend on the expected transmission delays of the balanced system. The 'transmission dilemma' is an important element of the balancing process.

This study is a part of an ongoing research in distributed load balancing systems. In the coming stages some of the restrictions of the model presented here will be released and distributed systems with other communication disciplines will be considered.

REFERENCES

(1) P. H. Enslow, Jr. "What is a Distributed Data Processing System ," Computer, Jenuary 1978, pp. 13-21.

(2) R. H. Echhouse, Jr., J. A. Stankovic, "Issues in Distributed Processing - An Overview of Two Workshops," Computer, January 1978, pp. 22 - 26.

(3) H. S. Stone, "Multiprocessor Scheduling with the Aid of Network Flow Algorithms," IEEE Trans. of Software Engin., Vol. SE-3, No. 1, January 1977, pp. 85-93.

(4) A. Kratzer, D. Hammerstrom, "A Study of Load Leveling," Proceedings of COMPCON, Fall 1980, pp. 647-654.

(5) Y. C. Chow, W. H. Kohler, "Dynamic Load Balancing in Homogeneous Two-Processor Distributed Systems," International Symposium on Computer Performance, Modeling, Measurement and Evaluation, Augast 1977, pp. 49-52.

(6) Y. Eran, M. Livny, M. Melman, "Modeling and Evaluation of a Tree Structured Network: a Case Study," Proceedings of MELECON 1981.

(7) L. Kleinrock, "Queueing Systems Vol. 1: Theory," Wiley, New york, 1975.

(8) R. M. Metcalfe, D. R. Boggs, "Ethernet: Distributed Packet Switching for Local Computer Networks," Xerox Research Report, csl-8Ø-2, 1980.

(9) A. K. Agrawala, R. M. Bryant, J. Agre, "Analysis of an Ether-Like Protocol," Computer Networking Symposium 197'7, pp. 104-111.

(1Ø) J. M. Mcquillan, I. Richer, E. C. Rosen, "The New Routing Algorithm for the ARPANET," IEEE Trans. on Communications, vol. COM-28, No. 5, May 1980, pp. 711-'719.