

Models for Dynamic Load Balancing in a Heterogeneous Multiple Processor System

YUAN-CHIEH CHOW AND WALTER H. KOHLER, MEMBER, IEEE

Abstract—Queueing models for a simple heterogeneous multiple processor system are presented, analyzed, and compared. Each model is distinguished by a job routing strategy which is designed to reduce the average job turnaround time by balancing the total load among the processors. In each case an arriving job is routed by a job dispatcher to one of m parallel processors. The job routing strategies are divided into two classes: deterministic and nondeterministic. The nondeterministic policies are described by state independent branching probabilities. For the deterministic policies, the next processor is chosen to minimize or maximize the expected value of a performance related criterion function.

The models with nondeterministic policies are analyzed using standard queueing network techniques. An approximate numerical method is introduced for analyzing two-processor heterogeneous models with deterministic routing policies and the performance of sample two-processor systems is compared. The comparison shows that a deterministic strategy that maximizes the expected throughput during the next interarrival interval gives the best performance. This adaptive policy is conjectured to be optimal among a class of policies.

Index Terms—Job routing, load balancing, multiple processor system, performance analysis, queueing models.

I. INTRODUCTION

ONE OF the major research issues associated with the design of distributed computer systems consisting of cooperating autonomous processors is task scheduling and its relationship to system performance [7]. Queueing networks are used extensively for modeling computer systems and have proved to be a powerful tool for performance analysis and prediction [4]. Standard network models account for the flow of customers through the network by means of fixed matrices of branching probabilities [1]. Models based on this approach are not capable of explicitly representing the job routing policies of a distributed system.

In this paper queueing models for a simple heterogeneous multiple processor system are presented, analyzed, and compared. Each model is distinguished by a job routing strategy which is designed to reduce the average job turnaround time by balancing the total load among the processors. In each case an arriving job is routed by a job dispatcher to

one of m parallel processors. Related systems have been studied by Kingman [10], Koenigsberg [13], McGregor and Boorstyn [16], and Livine and Boorstyn [15]. The job routing strategies are divided into two classes: deterministic and nondeterministic. The nondeterministic policies include those commonly used in queueing network models and defined by a matrix of branching probabilities. For the deterministic policies, the next processor is chosen to minimize or maximize the expected value of a performance related criterion function. This approach can explicitly represent the adaptive job routing strategies used in distributed systems.

In Section II nondeterministic routing strategies are reviewed and the performance of one common policy is analyzed using standard queueing theory techniques. In this case the job dispatcher chooses one of m processors based on the outcome of an independent trial in which the i th outcome has probability p_i . The p_i are assumed fixed and proportional to the processing power of processor i . Section III describes three models in which the job routing decision is designed to minimize or maximize the expected value of a performance related criterion function. In these cases the routing decision is a deterministic function of the current state and system parameters. The three policies studied are: minimum response time, minimum system time, and maximum throughput. It is shown that the state diagram for two-processor systems that use these policies is partitioned into two regions by a policy line that connects a sequence of balanced states. An approximate numerical solution method is developed for this class of systems in Section IV. The algorithm determines the equilibrium state probabilities for two-processor models in $O(|S|)$ time, where $|S|$ is the total number of system states. In Section V the job turnaround time of sample two-processor systems is compared as a function of system load. The comparison shows that the deterministic maximum throughput policy minimizes the average job turnaround time. This adaptive policy is conjectured to be optimal among the class of policies constrained to use the available system parameters and current state as the basis for the routing decision.

II. NONDETERMINISTIC ROUTING

Consider the simple heterogeneous multiple processor system model shown in Fig. 1. Each processor is modeled as a queue/server pair. This is a typical model for m independent processors [11] assumed to be characterized completely by their service rate μ_i . Jobs enter the system at mean rate λ and leave the system at mean rate μ .

MICROSOFT CORP.
EXHIBIT 1010

Manuscript received November 9, 1977; revised March 23, 1978, August 7, 1978, and December 11, 1978. This work was supported by the National Science Foundation under Research Grant MCS 76-03667.

Y.-C. Chow was with the Department of Electrical and Computer Engineering, University of Massachusetts, Amherst, MA 01003. He is now with the Department of Computer Science, Wright State University, Dayton, OH 45431.

W. H. Kohler is with the Department of Electrical and Computer Engineering, University of Massachusetts, Amherst, MA 01003.

(Duplicate first page for exhibit labeling, per 37 C.F.R. § 42.63(d)(2)(ii).)

At

apply.

Models for Dynamic Load Balancing in a Heterogeneous Multiple Processor System

YUAN-CHIEH CHOW AND WALTER H. KOHLER, MEMBER, IEEE

Abstract—Queueing models for a simple heterogeneous multiple processor system are presented, analyzed, and compared. Each model is distinguished by a job routing strategy which is designed to reduce the average job turnaround time by balancing the total load among the processors. In each case an arriving job is routed by a job dispatcher to one of m parallel processors. The job routing strategies are divided into two classes: deterministic and nondeterministic. The nondeterministic policies are described by state independent branching probabilities. For the deterministic policies, the next processor is chosen to minimize or maximize the expected value of a performance related criterion function.

The models with nondeterministic policies are analyzed using standard queueing network techniques. An approximate numerical method is introduced for analyzing two-processor heterogeneous models with deterministic routing policies and the performance of sample two-processor systems is compared. The comparison shows that a deterministic strategy that maximizes the expected throughput during the next interarrival interval gives the best performance. This adaptive policy is conjectured to be optimal among a class of policies.

Index Terms—Job routing, load balancing, multiple processor system, performance analysis, queueing models.

I. INTRODUCTION

ONE OF the major research issues associated with the design of distributed computer systems consisting of cooperating autonomous processors is task scheduling and its relationship to system performance [7]. Queueing networks are used extensively for modeling computer systems and have proved to be a powerful tool for performance analysis and prediction [4]. Standard network models account for the flow of customers through the network by means of fixed matrices of branching probabilities [1]. Models based on this approach are not capable of explicitly representing the job routing policies of a distributed system.

In this paper queueing models for a simple heterogeneous multiple processor system are presented, analyzed, and compared. Each model is distinguished by a job routing strategy which is designed to reduce the average job turnaround time by balancing the total load among the processors. In each case an arriving job is routed by a job dispatcher to

one of m parallel processors. Related systems have been studied by Kingman [10], Koenigsberg [13], McGregor and Boorstyn [16], and Livine and Boorstyn [15]. The job routing strategies are divided into two classes: deterministic and nondeterministic. The nondeterministic policies include those commonly used in queueing network models and defined by a matrix of branching probabilities. For the deterministic policies, the next processor is chosen to minimize or maximize the expected value of a performance related criterion function. This approach can explicitly represent the adaptive job routing strategies used in distributed systems.

In Section II nondeterministic routing strategies are reviewed and the performance of one common policy is analyzed using standard queueing theory techniques. In this case the job dispatcher chooses one of m processors based on the outcome of an independent trial in which the i th outcome has probability p_i . The p_i are assumed fixed and proportional to the processing power of processor i . Section III describes three models in which the job routing decision is designed to minimize or maximize the expected value of a performance related criterion function. In these cases the routing decision is a deterministic function of the current state and system parameters. The three policies studied are: minimum response time, minimum system time, and maximum throughput. It is shown that the state diagram for two-processor systems that use these policies is partitioned into two regions by a policy line that connects a sequence of balanced states. An approximate numerical solution method is developed for this class of systems in Section IV. The algorithm determines the equilibrium state probabilities for two-processor models in $O(|S|)$ time, where $|S|$ is the total number of system states. In Section V the job turnaround time of sample two-processor systems is compared as a function of system load. The comparison shows that the deterministic maximum throughput policy minimizes the average job turnaround time. This adaptive policy is conjectured to be optimal among the class of policies constrained to use the available system parameters and current state as the basis for the routing decision.

II. NONDETERMINISTIC ROUTING

Consider the simple heterogeneous multiple processor system model shown in Fig. 1. Each processor is modeled as a queue/server pair. This is a typical open queueing model for m independent processors [11], [12]. Processor i is assumed to be characterized completely by its mean service rate μ_i . Jobs enter the system at mean rate λ . Their interarri-

Manuscript received November 9, 1977; revised March 23, 1978, August 7, 1978, and December 11, 1978. This work was supported by the National Science Foundation under Research Grant MCS 76-03667.

Y.-C. Chow was with the Department of Electrical and Computer Engineering, University of Massachusetts, Amherst, MA 01003. He is now with the Department of Computer Science, Wright State University, Dayton, OH 45431.

W. H. Kohler is with the Department of Electrical and Computer Engineering, University of Massachusetts, Amherst, MA 01003.

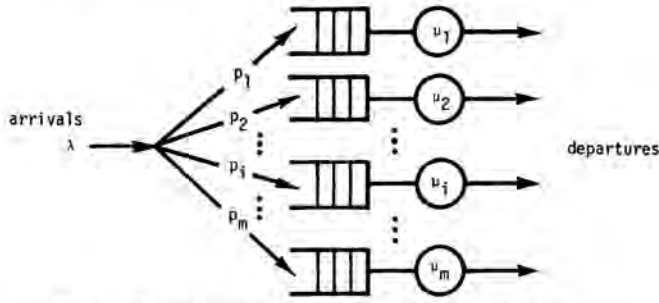


Fig. 1. Heterogeneous system with nondeterministic routing.

val times are modeled as a sequence of independent identically distributed random variables. The distribution is assumed to be exponential with a mean of $1/\lambda$. An arriving job is routed to processor i with probability p_i where the routing decision is based on the outcome of an independent trial. Since one of the m processors must be chosen, $\sum_{i=1}^m p_i = 1$. Each processor services its queue of jobs in first-come-first-served priority order. The processing time of a job on processor i is modeled as an independent exponentially distributed random variable with mean $1/\mu_i$.

Given the above description of the system components and workload, the behavior of the system can be described by a discrete-state continuous-time Markov process. The instantaneous state s of the system is a snapshot of the workload distribution among the system components and can be expressed as $s = (n_1, n_2, \dots, n_i, \dots, n_m)$, where n_i is the number of jobs in the i th queue including the job in service. As long as $p_i \neq 1$ for some i , the routing decision in the above model is nondeterministic. The p_i can be chosen arbitrarily or as a function of the system parameters and current state. A nondeterministic routing policy is state independent if the branching probabilities are determined independently of the system state.

Nondeterministic state-independent job routing policies are commonly used in queueing network models because efficient solution techniques have been developed. For open models with exponential arrival and service distributions, the state equilibrium probabilities have a simple product form solution [9]. In the case of closed models, the local balance property exists for systems with exponential service distributions or special priority rules and a product form solution can again be found [1]–[3]. If the p_i are fixed, the system in Fig. 1 can be decomposed and analyzed as m independent $M/M/1$ systems, where system i has arrival rate λp_i and service rate μ_i . As long as $\lambda p_i < \mu_i$, the i th system is unsaturated and the mean turnaround time for jobs serviced by processor i is then $1/(\mu_i - \lambda p_i)$ [11]. Furthermore, since p_i is the proportion of jobs serviced by processor i , the mean turnaround time TT for the overall system can be expressed as the weighted sum

$$TT = \sum_{i=1}^m p_i \left(\frac{1}{\mu_i - \lambda p_i} \right). \quad (1)$$

One natural choice is to fix p_i in proportion to the processing speed of processor i , i.e., $p_i = \mu_i / \sum_{j=1}^m \mu_j$. Although this *proportional branching policy* may not mini-

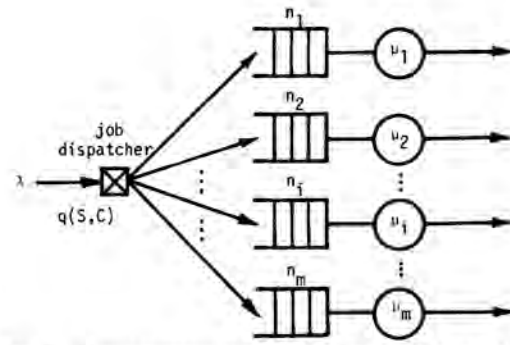


Fig. 2. Heterogeneous system with deterministic routing.

mize the job turnaround time, it does guarantee that each queue will be unsaturated as long as $\lambda < \sum_{j=1}^m \mu_j$. From the definition of p_i and the above inequality it follows that

$$\lambda p_i = \lambda \frac{\mu_i}{\sum_{j=1}^m \mu_j} = \mu_i \frac{\lambda}{\sum_{j=1}^m \mu_j} < \mu_i.$$

Using (1) and the definition of p_i , the mean turnaround time for the proportional routing policy TT_{PR} can be expressed as

$$\begin{aligned} TT_{PR} &= \sum_{i=1}^m p_i \left(\frac{1}{\mu_i - \lambda p_i} \right) \\ &= \frac{m}{\sum_{j=1}^m \mu_j - \lambda}. \end{aligned} \quad (2)$$

III. DETERMINISTIC ROUTING

Consider the multiple processor heterogeneous system model in Fig. 2. This model is the same as Fig. 1 except for the inclusion of the job dispatcher. If an arriving job finds the system in state $s = (n_1, n_2, \dots, n_m)$, then the job will be dispatched (routed) to queue $q(s, C)$, where $q(s, C)$ is determined by a system criterion function C . The notation $C(n_1, n_2, \dots, n_i + 1, \dots, n_m)$ is used to denote the value of the function when the job is sent to the i th queue. The job dispatching strategy is deterministic since an arriving job is always routed to a processor in accordance with the deterministic criterion function of the job dispatcher. Three state-dependent routing policies will be investigated: the minimum response time policy, the minimum system time policy, and the maximum throughput policy. Criterion function C will be expressed in terms of the system parameters. In Section IV it will be shown that two-processor heterogeneous systems with these routing policies can be solved approximately using an efficient recursive solution technique. The performance of these systems is compared in Section V.

A. Minimum Response Time Policy (R)

The minimum response time policy routes an arriving job to the system that offers the least expected turnaround time. A generalized version of this policy has recently been studied

by Foschini [8] in connection with dynamic routing in packet switched communications networks. The policy can be stated formally as follows.

Assume the system is in state $s = (n_1, n_2, \dots, n_m)$.

1) An arriving job is sent to queue i if the queue length to service rate ratio $(n_i + 1)/\mu_i = \min \{(n_k + 1)/\mu_k | k = 1, 2, \dots, m\}$.

2) If the minimum ratio is not unique, the job dispatcher selects from among the ties the one with maximum service rate μ_i .

Note that we assume $\mu_i \neq \mu_j$ for $i \neq j$. Condition 2) is one of many possible ways to resolve a tie and it will be used for each deterministic policy studied. The minimum response time policy is based on an individual user's view of how to improve performance, since $((n_k + 1)/\mu_k)$ is the expected response time of the next job in system k .

The following notation will be used to represent the minimum response time policy.

Minimum Response Time Policy (R): Let

$$R(n_1, n_2, \dots, n_k + 1, \dots, n_m) \triangleq \frac{n_k + 1}{\mu_k}$$

$$R_{\min}(s) \triangleq \min \{R(n_1, n_2, \dots, n_k + 1, \dots, n_m) | k = 1, 2, \dots, m\}$$

$$Q(s, R) \triangleq \{k | R_{\min}(s) = R(n_1, n_2, \dots, n_k + 1, \dots, n_m)\}.$$

$$\begin{aligned} T(n_1, n_2, \dots, n_m) &= E[\text{time to service every job}] \\ &= E[\text{time to service the next job}] \\ &\quad + E[\text{time to service the remaining jobs}] \\ &= \frac{1}{\sum_{j=1}^m \mu_j} + \sum_{k=1}^m \left[\frac{\mu_k}{\sum_{j=1}^m \mu_j} T(n_1, n_2, \dots, n_k - 1, \dots, n_m) \right], \end{aligned}$$

$$T(0, 0, \dots, n_k, \dots, 0, 0) = \frac{n_k}{\mu_k}.$$

Then

$$q(s, R) = i, \quad \text{where } i \in Q(s, R)$$

and

$$\mu_i < \mu_j \quad \forall j \in Q(s, R), \quad j \neq i.$$

Since the departure transitions in the state-transition-rate diagram are independent of the routing policy, it is convenient to simplify the diagram by showing only the arrival transitions. Fig. 3 shows a grid which represents the simplified state diagram for a two-processor system with $\mu_1 = 2$ and $\mu_2 = 1$. Each intersection represents a state (i, j) . The arrows indicate transitions from state (i, j) to (i', j') of rate λ due to an arrival. The departure transitions from state (i, j) to $(i, j - 1)$ and $(i - 1, j)$ of rates μ_1 and μ_2 are omitted. It will be shown in Section IV that this state diagram satisfies

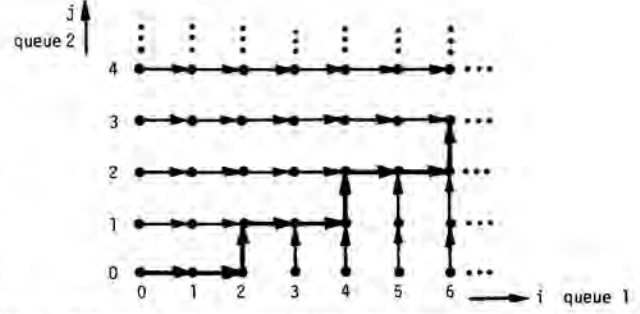


Fig. 3. Simplified state diagram for minimum response time policy when $\mu_1 = 2$ and $\mu_2 = 1$.

special properties and can be solved approximately using an efficient recursive method.

B. Minimum System Time Policy (T)

The minimum system time policy is motivated by two goals, to find an optimal routing strategy and to model parallel processing. Our analysis shows that although the policy does improve system performance, it is not an optimal routing strategy. Given that the system is in state $s = (n_1, n_2, \dots, n_m)$, the system time $T(n_1, n_2, \dots, n_m)$ is defined as the expected time to complete every job already in the system. This definition can be expressed recursively as follows:

The term $1/\sum_{j=1, n_j \neq 0}^m \mu_j$ represents the average time until the next job completes its service given that the system is in state $(n_1, n_2, \dots, n_m)$. This follows from the assumption that each server has an exponential service distribution. The expression $\mu_k/\sum_{j=1, n_j \neq 0}^m \mu_j$ is the probability that the next departing job is from queue k . If a job departs from queue k , then the average remaining system time can be defined recursively using the first equation. The second equation defines the boundary conditions for the recurrence relation. If $T(n_1, n_2, \dots, n_m)$ is chosen as the criterion function, then the minimum system time policy can be expressed formally as follows.

Minimum System Time Policy (T): Let

$$T_{\min}(s) \triangleq \min \{T(n_1, n_2, \dots, n_k + 1, \dots, n_m) | k = 1, 2, \dots, m\}$$

$$Q(s, R) \triangleq \{k | T_{\min}(s) = T(n_1, n_2, \dots, n_k + 1, \dots, n_m)\}.$$

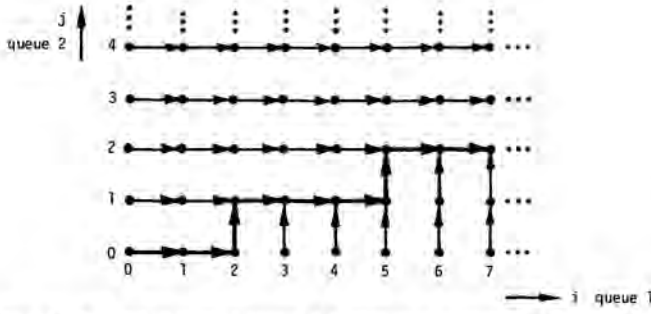


Fig. 4. Simplified state diagram for minimum system time policy when $\mu_1 = 2$ and $\mu_2 = 1$.

Then $q(s, R) = i$, where $i \in Q(s, R)$ and $\mu_i < \mu_j \forall j \in Q(s, R), j \neq i$.

For the special case of a two-processor system,

$$T(n_1, n_2) = \frac{1}{\mu_1 + \mu_2} + \frac{\mu_1}{\mu_1 + \mu_2} T(n_1 - 1, n_2) \\ + \frac{\mu_2}{\mu_1 + \mu_2} T(n_1, n_2 - 1), \\ T(n_1, 0) = \frac{n_1}{\mu_1},$$

and

$$T(0, n_2) = \frac{n_2}{\mu_2}.$$

The simplified state diagram for this policy when $\mu_1 = 2$ and $\mu_2 = 1$ is given by Fig. 4.

C. Maximum Throughput Policy (TP)

The minimum response time and minimum system time policies are functions of the system queue lengths and processor service rates only. The job dispatcher may also have information available on the job arrival rate. In this section, it is shown how this additional information can be used to improve the performance of the system. Consider the equivalent single queue model for the multiple processor system as shown in Fig. 5 where μ_e is the equivalent service rate. μ_e depends on the workload distribution among the queues, although this distribution is not explicitly shown in the composite queue diagram. The function of the job dispatcher is to update the workload distribution after an arrival so that the minimum average turnaround time can be achieved. It seems plausible that the job turnaround time will decrease if the service rate can be increased. The equivalent average service rate μ_e during an interarrival time is defined as the average system throughput during this interval. The workload distribution can be updated by the dispatcher only when a job arrives. The maximum throughput job routing policy is a strategy that assigns each arrival to the queue that will maximize the expected throughput of the system during the next interarrival period.

The throughput of the system is the sum of the individual throughputs of each processor in the system. The average

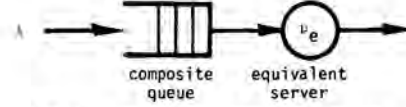


Fig. 5. Equivalent single queue model.

throughput $TP_i(n_i)$ of processor i during an interarrival period can be derived as follows. Let

$Q_{ik} = \Pr$ [exactly k jobs leave processor i during an interarrival period of length τ | n_i jobs present initially].

Since the service distribution of processor i is exponential, the departure process is Poisson and the probability Q_{ik} is given by

$$Q_{ik} = \frac{(\mu_i \tau)^k e^{-\mu_i \tau}}{k!}, \quad 0 \leq k < n_i \\ Q_{in_i} = 1 - \sum_{k=1}^{n_i-1} Q_{ik} = \sum_{k=n_i}^{\infty} \frac{(\mu_i \tau)^k e^{-\mu_i \tau}}{k!}.$$

The expected throughput of processor i during an interarrival interval of length τ is the average number of jobs that are serviced per unit time and is equal to $\sum_{k=1}^{n_i} k \cdot Q_{ik} / \tau$, where n_i is the initial number of jobs in queue i . Because of the Poisson arrival assumption, the probability that the interarrival time is τ is given by $f(\tau) = \lambda e^{-\lambda \tau}$. Thus, the average throughput of processor i can be expressed as

$$TP_i(n_i) = \int_0^{\infty} \frac{\sum_{k=1}^{n_i} k Q_{ik}}{\tau} f(\tau) d\tau \\ = \int_0^{\infty} \frac{\sum_{k=1}^{n_i-1} k Q_{ik}}{\tau} f(\tau) d\tau + \int_0^{\infty} \frac{n_i Q_{in_i}}{\tau} f(\tau) d\tau \\ = \sum_{k=1}^{n_i-1} \lambda \left(\frac{\mu_i}{\lambda + \mu_i} \right)^k \int_0^{\infty} \frac{(\lambda + \mu_i)^k \tau^{k-1}}{(k-1)!} e^{-(\lambda + \mu_i)\tau} d\tau \\ + n_i \lambda \sum_{k=n_i}^{\infty} \left(\frac{\mu_i}{\lambda + \mu_i} \right)^k \frac{1}{k} \\ \cdot \int_0^{\infty} \frac{(\lambda + \mu_i)^k \tau^{k-1}}{(k-1)!} e^{-(\lambda + \mu_i)\tau} d\tau \\ = \sum_{k=1}^{n_i-1} \lambda \left(\frac{\mu_i}{\lambda + \mu_i} \right)^k \\ + n_i \lambda \sum_{k=n_i}^{\infty} \left(\frac{\mu_i}{\lambda + \mu_i} \right)^k \frac{1}{k}.$$

Now let $X = \mu_i / (\lambda + \mu_i)$ and simplify the expression as follows:

$$TP_i(n_i) = \lambda \left[\sum_{k=1}^{n_i-1} X^k + n_i \sum_{k=n_i}^{\infty} \frac{X^k}{k} \right] \\ = \lambda \left[\sum_{k=1}^{n_i-1} \left(X^k - \frac{n_i}{k} X^k \right) + n_i \sum_{k=1}^{\infty} \frac{X^k}{k} \right] \\ = \lambda \left[\sum_{k=1}^{n_i-1} \left(1 - \frac{n_i}{k} \right) X^k - n_i \ln(1 - X) \right].$$

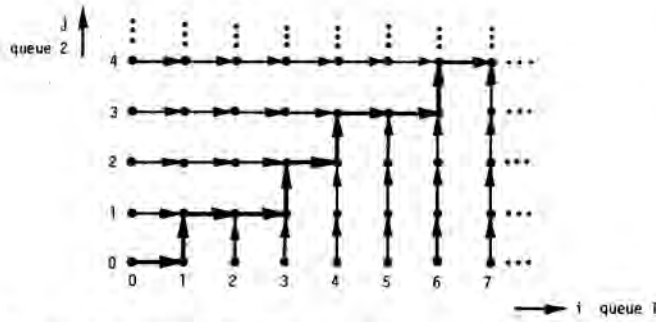


Fig. 6. State diagram for maximum throughput policy when $\lambda = 1.5$, $\mu_1 = 2$, and $\mu_2 = 1$.

Given that the system is in state $(n_1, n_2, \dots, n_m)$, the overall average throughput $TP(n_1, n_2, \dots, n_m)$ during the next interarrival period is the sum of all the individual $TP_i(n_i)$. This becomes

$$TP(n_1, n_2, \dots, n_m) = \sum_{i=1}^m \lambda \left[\sum_{k=1}^{n_i-1} \left(1 - \frac{n_i}{k} \right) \left(\frac{\mu_i}{\lambda + \mu_i} \right)^k - n_i \ln \left(\frac{\lambda}{\lambda + \mu_i} \right) \right]$$

The maximum throughput routing policy is defined formally as follows.

Maximum Throughput Policy (TP): Let

$$TP_{\max}(s) \triangleq \max \{ TP(n_1, n_2, \dots, n_k + 1, \dots, n_m) \mid k = 1, 2, \dots, m \}$$

$$Q(s, R) \triangleq \{ k \mid TP_{\max}(s) = TP(n_1, n_2, \dots, n_k + 1, \dots, n_m) \}.$$

Then

$$q(s, R) = i, \quad \text{where } i \in Q(s, R)$$

and

$$\mu_i < \mu_j \quad \forall j \in Q(s, R), \quad j \neq i.$$

Fig. 6 shows the simplified state diagram for a two-processor system. The arrival transitions, which are indicated by arrows, are a function of λ , μ_1 , μ_2 , n_1 , and n_2 . The major difference between the maximum throughput policy and the other two policies is that it takes into account the average arrival rate information λ . In Section V it will be shown numerically that this policy minimizes the average job turnaround time among the policies studied for a sample two-processor system.

IV. RECURSIVE SOLUTION TECHNIQUE

In this section an efficient recursive technique for analyzing two-processor heterogeneous systems with special properties is developed. (It is a generalization of the approach used in [6] for homogeneous systems.) The models with deterministic state-dependent routing policies discussed in Section III will be shown to satisfy these properties. The efficient analysis of such models thus becomes possible. This method will be used to compare the performance of sample systems in Section V.

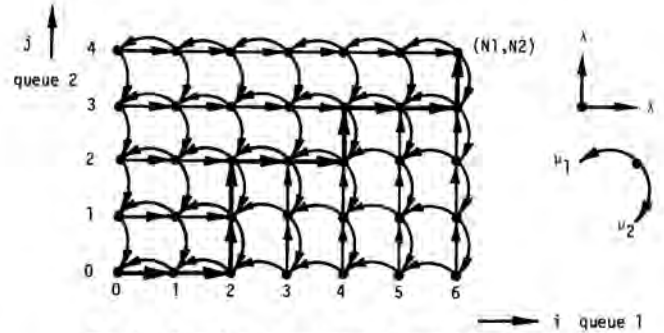


Fig. 7. Sample two-processor heterogeneous system.

Consider the state-transition-rate diagram in Fig. 7 representing a closed two-processor heterogeneous system with arrival rate λ and service rates μ_1 and μ_2 , $\mu_1 > \mu_2$. The job dispatcher assigns the next arriving job to one of the two queues depending upon a predefined deterministic criterion. Every well defined job routing policy partitions the state diagram into three classes of states. Fig. 7 is an example of one such partition. The chain of dark arrival transitions with rate λ will be called the *policy line* of the system. All arrival transitions move the system state toward the policy line. The states connected by the policy line are called *balanced states*. The states above the policy line are termed *upper unbalanced states* and those below *lower unbalanced states*. If the equilibrium state probabilities P_{ij} of the system exist, the infinite state diagram can be truncated to an $N_1 + 1$ by $N_2 + 1$ rectangle, with the policy line ending at state (N_1, N_2) . This approximation will introduce negligible error if the system is unsaturated and N_1 and N_2 are reasonably large, since P_{ij} will rapidly approach zero for $i \gg 0$ and $j \gg 0$. Alternately, Fig. 7 can be viewed as the *exact model* of a closed system with a total of $N_1 + N_2$ circulating jobs.

The recursive solution method for two-processor systems starts by solving for the prenormalized state probabilities of the outer boundary states along column N_1 or row N_2 in terms of some arbitrary initial value assigned to state (N_1, N_2) . The global balance equation (GBE) of state (N_1, N_2) is then used to solve for the prenormalized probability of state $\text{prev}(N_1, N_2)$, where function $\text{prev}(i, j)$ maps balanced state (i, j) into its previous balanced state on the policy line. For example, $\text{prev}(N_1, N_2) = (N_1, N_2 - 1)$ in Fig. 7. The method proceeds recursively, solving states along a column or a row until $\text{prev}(i, j) = (0, 0)$. All previously computed probabilities are then normalized.

This recursive method is described by the iterative algorithm given below and illustrated by a simple example in the Appendix. X is a variable representing a temporarily unknown boundary state probability and GBE is used to denote a global balance equation.

1. Set $(i, j) = (N_1, N_2)$.
2. Set $P_{ij} = \text{CONST}$ (some arbitrary value).
3. For $K = N_1 + N_2$ down to 1 do steps 4 through 15.
4. Set $(i', j') = \text{prev}(i, j)$.
5. If $i = i'$, then do steps 6 through 8.
6. Set $P_{0j} = X$.

7. For $I = 1$ to $i - 1$ use the GBE of state $(I - 1, j)$ to find an expression for $P_{I,j}$ as a function of X .
8. Use the GBE of state $(i - 1, j)$ to solve for X .
9. If $j = j'$ then do steps 10 through 12.
10. Set $P_{i0} = X$.
11. For $J = 1$ to $j - 1$ use the GBE of state $(i, J - 1)$ to find an expression for $P_{i,J}$ as a function of X .
12. Use the GBE of state $(i, j - 1)$ to solve for X .
13. Use the GBE of state (i, j) to solve for $P_{i,j}$.
14. Set $(i, j) = (i', j')$.
15. Continue.
16. Normalize all P_{ij} .

Since the global balance equation of each state except (N_1, N_2) and $(0, 0)$ is used exactly once and in each case the time required can be bounded by a constant that is independent of N_1 and N_2 , the overall time complexity is proportional to the total number of states $|S| = (N_1 + 1)(N_2 + 1)$.

The important characteristic of the state-transition-rate diagram in Fig. 7 that enables it to be solved efficiently is that each upper unbalanced state (i, j) transits right to state $(i + 1, j)$ upon an arrival, while each lower unbalanced state (i, j) goes up to state $(i, j + 1)$. This means that each upper unbalanced state has an excess of jobs in queue 2 and therefore jobs are dispatched to queue 1 whenever possible. The complement is true for the lower unbalanced states.

When discussing two-processor systems it is more convenient to replace the max and min functions used previously in the definitions of job routing policies with a relational operator R . Let $C(i + 1, j)$ and $C(i, j + 1)$ be the values of the criterion function after a job is dispatched to queue 1 and queue 2, respectively, and assume $\mu_1 > \mu_2$. The routing strategy of $\max \{C(i + 1, j), C(i, j + 1)\}$, with ties broken in favor of the faster processor, can then be described as

$$q(s, C) = \begin{cases} 1, & \text{if } C(i + 1, j) R C(i, j + 1) = \text{true} \\ 2, & \text{otherwise,} \end{cases}$$

where R is $\geq$.

The special properties of a job routing policy that enable the above recursive solution technique to be used for the two-processor heterogeneous systems will now be stated formally.

Property 1: The job routing policy is deterministic, i.e., if the system is in state (i, j) just prior to an arrival, it moves to state $(i + 1, j)$ if $C(i + 1, j) R C(i, j + 1)$ is true. Otherwise, the system goes from state (i, j) to state $(i, j + 1)$.

Definition: The *policy line* of a routing strategy is the continuous chain of arrival transitions starting at state $(0, 0)$ given that no departure occurs.

Property 2: The policy line partitions the states of the state-transition-rate diagram into two regions. For each state (i, j) in the upper unbalanced region

$$C(i + 1, j + n) R C(i, j + n + 1) = \text{true} \quad \forall n \geq 0$$

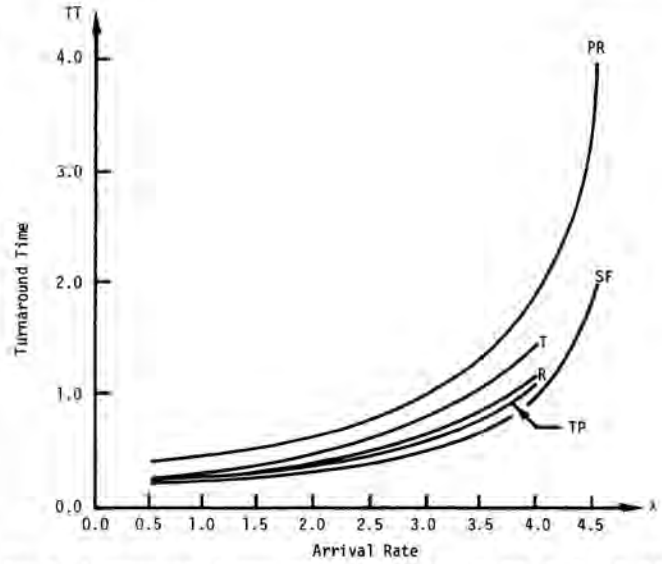


Fig. 8. Comparison of turnaround time for Models PR, T, R, TP, SF when $\mu_1 = 4$ and $\mu_2 = 1$.

while for each state (i, j) in the lower unbalanced region

$$C(i + n, j + 1) R C(i + n + 1, j) = \text{true} \quad \forall n \geq 0.$$

Chow has shown [5] that any two-processor system with a policy that satisfies Property 1 and Property 2 can be solved using the recursive solution method and that each of the deterministic policies studied in Section III (minimum response time, minimum system time, maximum throughput) is such a policy. The generalization of this technique for three or more processors does not appear to be straightforward.

V. COMPARISON OF PERFORMANCE

The average job turnaround times of sample two-processor heterogeneous models are compared in this section. These models are labeled as follows:

- Model PR: state independent routing with proportional branching probabilities,
- Model T: minimum system time policy,
- Model R: minimum response time policy,
- Model TP: maximum throughput policy,
- Model SF: single fast processor.

In Section II it was shown that the average turnaround time of Model PR can be expressed in closed form as $2/(\mu_1 + \mu_2 - \lambda)$. Models T, R, and TP are solved by using the recursive solution technique. The computed state equilibrium probabilities are then used to evaluate the average job turnaround time. Model SF is a standard $M/M/1$ system with service rate $\mu_1 + \mu_2$, and the average turnaround time is given by

$$TT_{SF} = \frac{1}{\mu_1 + \mu_2 - \lambda}.$$

As expected, the maximum throughput policy achieves the best average job turnaround time among the multiple

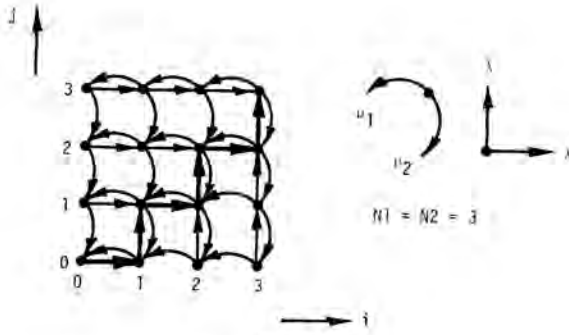


Fig. 9.

processor models. Using TT_k to denote the average turnaround time of model K , the analysis of the two-processor models when $\mu_1 = 4$ and $\mu_2 = 1$ indicates that

$$TT_{PR} \geq TT_T \geq TT_R \geq TT_{TP} \geq TT_{SF}$$

for each choice of λ . Fig. 8 is a plot of the data. Data points for Models T , R , and TP are not shown when $\lambda = 4.5$. This is due to the difficulty of accurately approximating an open system by a small finite-state diagram when the $\lambda/(\mu_1 + \mu_2)$ ratio approaches one.

Several conclusions can be drawn from the above comparison. Under light load conditions, Models T , R , and TP closely approximate the performance of the single fast processor Model SF . The performance of Models T and R degrade more than that of Model TP as λ increases. This is due to the fact that the minimum system time (Model T) and minimum response time (Model R) policies fail to consider the effect of the arrival rate. These two policies tend to schedule too many jobs to the fast processor while leaving the slower processor idle. Although this strategy is effective for low arrival rates, the system achieves better performance under heavy load conditions when the overall processing power is utilized. The maximum throughput policy is adaptive to all system information available to the job dispatcher. In the limit as $\lambda \rightarrow \mu_1 + \mu_2$ the performance of the three deterministic policies tends to approach the single fast processor, which has a two-to-one advantage over the static proportional routing policy. Foschini's [8] recent results using the diffusion approximation verify this behavior for the minimum response time policy.

VII. DISCUSSION

Queueing models for a simple heterogeneous multiple processor system have been presented, analyzed, and compared. Those models incorporating adaptive job routing strategies reduced job turnaround time by balancing the total load among the processors. It has been shown independently that under certain assumptions the reduction factor in an m processor system approaches $1/m$ [8]. The maximum throughput policy is conjectured to be optimal among all policies that are based on the system state, average arrival rate, and average service rate information.

While the commonly used local balance queueing network models are based on static nondeterministic bran-

ching policies, real systems are likely to employ adaptive deterministic policies. It has been shown [5] that dynamic load balancing reduces the variance of the arrival distribution, changing it from exponential to hypoexponential. The present results demonstrate that models incorporating adaptive policies can be analyzed and that their performance differs significantly from the standard models. This also indicates that local balance queueing models may be inadequate for modeling distributed systems employing dynamic load balancing policies.

APPENDIX

DEMONSTRATION OF RECURSIVE SOLUTION TECHNIQUE

Solution steps for a sample system using the algorithm in Section IV follow (see Fig. 9).

- 1) Set $P_{33} = 1$.
- 2) Set $P_{03} = X$.
- 3) Use the GBE for state $(0, 3)$ to determine P_{13} ,

$$(\mu_2 + \lambda)P_{03} = \mu_1 P_{13}$$

$$P_{13} = \frac{1}{\mu_1} (\mu_2 + \lambda)X.$$

- 4) Use the GBE for state $(1, 3)$ to determine P_{23} ,

$$(\mu_1 + \mu_2 + \lambda)P_{13} = \lambda P_{03} + \mu_1 P_{23}$$

$$P_{23} = \frac{1}{\mu_1} \left[(\mu_1 + \mu_2 + \lambda) \frac{\mu_2 + \lambda}{\mu_1} X - \lambda X \right]$$

$$= \frac{\mu_1 \mu_2 + (\mu_2 + \lambda)^2}{\mu_1^2} X.$$

- 5) Use the GBE for state $(2, 3)$ to solve for X ,

$$(\mu_1 + \mu_2 + \lambda)P_{23} = \lambda P_{13} + \mu_1 P_{33}$$

$$\frac{(\mu_1 + \mu_2 + \lambda)[\mu_1 \mu_2 + (\mu_2 + \lambda)^2]}{\mu_1^2} X = \frac{\lambda}{\mu_1} (\mu_2 + \lambda)X + \mu_1$$

$$X = \frac{\mu_1^3}{(\mu_1 + \mu_2 + \lambda)[\mu_1 \mu_2 + (\mu_2 + \lambda)^2] - \lambda \mu_1 (\mu_2 + \lambda)}$$

- 6) Substitute the value of X into equations in steps 2), 3), 4) to obtain numerical values for P_{03} , P_{13} , and P_{23} .
- 7) Use the GBE for state $(3, 3)$ to determine P_{32} ,

$$(\mu_1 + \mu_2)P_{33} = \lambda P_{23} + \lambda P_{32}$$

$$P_{32} = \frac{1}{\lambda} [\mu_1 + \mu_2 - \lambda P_{23}].$$

- 8) P_{32} is now known since we have the numerical value for P_{23} .
- 9) Set $P_{30} = X$.
- 10) Use the GBE for state $(3, 0)$ to determine P_{31} ,

$$(\mu_1 + \lambda)P_{30} = \mu_2 P_{31}$$

$$P_{31} = \frac{1}{\mu_2} (\mu_1 + \lambda)X.$$

11) Use the GBE of state (3, 1) to solve for X ,

$$(\mu_1 + \mu_2 + \lambda)P_{31} = \lambda P_{30} + \mu_2 P_{32}$$

$$(\mu_1 + \mu_2 + \lambda) \frac{\mu_1 + \lambda}{\mu_2} X = \lambda X + \mu_2 P_{32}$$

$$X = \frac{\mu_2^2 P_{32}}{(\mu_1 + \mu_2 + \lambda)(\mu_1 + \lambda) - \lambda \mu_2}$$

$$= \frac{\mu_2^2 P_{32}}{\mu_1 \mu_2 + (\mu_1 + \lambda)^2}$$

12) Substitute the value of X into equations in steps 9) and 10) to obtain numerical values for P_{30} and P_{31} .

13) Use the GBE for state (3, 2) to determine P_{22} ,

$$(\mu_1 + \mu_2 + \lambda)P_{32} = \lambda P_{22} + \mu_2 P_{33} + \lambda P_{31}$$

$$P_{22} = \frac{1}{\lambda} [(\mu_1 + \mu_2 + \lambda)P_{32} - \mu_2 P_{33} - \lambda P_{31}]$$

14) We now have the numerical value for P_{22} since P_{33} , P_{32} , and P_{31} are known.

15) We have reduced the 4×4 state diagram to 3×3 with P_{22} known. The same process may be repeated to solve all other state probabilities.

16) Normalize all P_{ij} by setting

$$P_{ij} \leftarrow \frac{P_{ij}}{\sum_{k=0}^3 \sum_{l=0}^3 P_{kl}}$$

ACKNOWLEDGMENT

The authors would like to acknowledge and thank their colleague Prof. D. F. Towsley for his many helpful discussions during the course of this research and the referees for their careful reading and corrections of the original manuscript.

REFERENCES

- [1] F. Baskett, K. M. Chandy, R. R. Muntz, and F. G. Palacios, "Open, closed, and mixed networks of queues with different classes of customers," *J. Ass. Comput. Mach.*, vol. 22, pp. 248-260, Apr. 1975.
- [2] J. P. Buzen, "Computational algorithms for closed queueing networks with exponential server," *Commun. Ass. Comput. Mach.*, vol. 16, pp. 527-531, Sept. 1973.
- [3] K. M. Chandy, "The analysis and solutions for general queueing networks," in *Proc. Sixth Annual Princeton Conf.*, 1972, pp. 224-228.
- [4] K. M. Chandy and M. Reiser, Eds., *Computer Performance*. New York: North-Holland, 1977.
- [5] Y. C. Chow, "Queueing models for distributed computer systems," Ph.D. dissertation, Dep. Comput. Inform. Sci., Univ. of Massachusetts, Amherst, Aug. 1977.
- [6] Y. C. Chow and W. H. Kohler, "Dynamic load balancing in homogeneous two-processor distributed systems," in *Computer Performance*, K. M. Chandy and M. Reiser, Eds. New York: North-Holland, 1977.

- [7] P. H. Enslow, Jr., "Distributed data processing systems: Some definitional and research issues," Brown Univ. Distributed Processing Workshop, Aug. 1977.
- [8] G. J. Foschini, "On heavy traffic diffusion analysis and dynamic routing in packet switched networks," in *Computer Performance*, K. M. Chandy and M. Reiser, Eds. New York: North-Holland, 1977.
- [9] J. R. Jackson, "Jobshop-like queueing systems," *Management Sci.*, vol. 10, pp. 131-142, Oct. 1963.
- [10] J. F. C. Kingman, "Two similar queues in parallel," *Ann. Math. Statist.*, vol. 32, pp. 1314-1323, 1961.
- [11] L. Kleinrock, *Queueing Systems, Vol. I: Theory*. New York: Wiley, 1975.
- [12] —, *Queueing Systems, Vol. II: Computer Applications*. New York: Wiley, 1976.
- [13] E. Koenigsberg, "On jockeying in queues," *Management Sci.*, vol. 12, pp. 412-436, Jan. 1966.
- [14] J. D. C. Little, "A proof of the queueing formula $L = \lambda W$," *Oper. Res.*, vol. 9, pp. 383-387, 1961.
- [15] A. Livine and R. R. Boorstyn, "On a technique for adaptive routing," presented at Nat. Telecommun. Conf., Dallas, TX, Nov. 1976.
- [16] P. V. McGregor and R. R. Boorstyn, "Optimal load sharing in a computer network," in *Proc. 1975 Int. Conf. Commun.*, vol. III, June 1975, pp. 41-14-19.
- [17] D. F. Towsley, "Local balance models of computer systems," Ph.D. dissertation, Univ. of Texas, Austin, Dec. 1975.
- [18] —, "Queueing network models with state dependent routing," submitted to *J. Ass. Comput. Mach.*, June 1977.



Yuan-Chieh Chow was born in Taiwan, China, on December 24, 1945. He received the B.S. degree in electronic engineering in 1968 from National Chiao-Tung University, Taiwan, Republic of China, the M.S. degree in electrical engineering in 1971 from University of Cincinnati, Cincinnati, OH, and the M.S. and Ph.D. degrees in computer science from University of Massachusetts, Amherst, in 1974 and 1977, respectively.

From 1971 to 1974, he was a Staff Assistant at the University of Massachusetts. He has been an Assistant Professor in Computer Science at Wright State University since 1977. His current interests include computer organization and system evaluation.

Dr. Chow is a member of the Association for Computing Machinery.



Walter H. Kohler (S'66-M'73) was born in Dover, NJ, on June 2, 1945. He received the B.S.E.E., M.S.E., M.A., and Ph.D. degrees in electrical engineering from Princeton University, Princeton, NJ, in 1967, 1968, 1971, and 1972, respectively.

From 1968 to 1969 he was a member of the Technical Staff at Bell Telephone Laboratories, Holmdel, NJ. Since 1972 he has been at the University of Massachusetts, Amherst, where he is currently an Associate Professor of Electrical and Computer Engineering. He is engaged in research and teaching in the areas of algorithms and computer systems design and evaluation.

Dr. Kohler is a member of the Association for Computing Machinery and Eta Kappa Nu.