Dataflow modelling in distributed diagnostic processing systems: a closed queuing network model with multiple servers

Vidhyacharan Bhaskar, Kondo Hloindo Adjallah, Laurie Joiner

▶ **To cite this version:**

Vidhyacharan Bhaskar, Kondo Hloindo Adjallah, Laurie Joiner. Dataflow modelling in distributed diagnostic processing systems: a closed queuing network model with multiple servers. International Journal of Pure and Applied Mathematics, 2005, 19 (1), pp.25-42. hal-03618693

HAL Id: hal-03618693
https://hal.science/hal-03618693

Submitted on 27 Mar 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons CC0 - Public Domain Dedication 4.0 International License

Datafl w Modelling in Distributed Diagnostic-Processing Systems A Closed Queuing Network Model Appoach with Multiple Servers

V. Bhaskar[1], K.H. H. Adjallah[2] and L. Joiner[3]

[1]Departement Genie des Systemes d'Information et Mecaniques
[2]Institute of Computer Science and Engineering of Troyes
Universite de Technologie de Troyes
10010 Troyes Cedex, FRANCE
[3]Department of Electrical Engineering and Computer Engineering
University of Alabama Huntsville, 35899 AL, USA

Abstract— **In this paper, a closed queuing network model with multiple servers has been proposed to model dataflow in distributed diagnostic-processing systems. Multi-threading is useful in reducing the latency by switching among a set of threads in order to improve the processor utilization. Two sets of processors, synchronization and execution processors exist. Synchronization processors handle load/store operations and execution processors handle arithmetic/logic and control operations. A closed queuing network model is suitable for large number of job arrivals. The normalization constant is derived using a recursive algorithm for the given model. Performance measures such as average response times and average system throughput are derived and plotted against the total number of processors in the closed queuing network model. Other important performance measures like processor utilizations, average queue lengths, average waiting times and relative utilizations are also derived.**

Index Terms— **Synchronization and Execution processors, Multi-programming, Queue lengths, Response times, Utilizations, Throughputs.**

I. Introduction

In an open-queuing network model, a job is scheduled to the main-memory and is able to compete for active resources such as synchronization and execution processors immediately on its arrival [1]. In practice, the number of main-memory partitions are limited. So, the existence of an additional queue is necessary. This queue is called a job-scheduler queue [1]. However, such a network is said to be multiple-resources holding. This is because a job cannot simultaneously hold main-memory and an active device. Such a network cannot be solved by product-form methods.

If the external arrival rate is low, then the probability that the job has to wait in the scheduler queue is low. So, the open-queuing network model is a good solution when the arrival rate is low. In other words, an open-queuing network model is a light-load approximation.

If the external arrival rate is high, the probability that there is at least one customer in the job-scheduler queue is very high. The departure of a job from the active-set immediately triggers the scheduling of an already waiting job into main-memory. Thus, a closed-queuing network model becomes imminent.

Input to the synchronization processor is in the form of threads and comprises a statistically determined sequence of RISC-style instructions [2]. Threads (sequence of instructions) are scheduled dynamically to be executed by execution processors. The threads have a bounded execution time [2]. Our model also represents a distributed shared memory system (DSM) model in which all processors share a common address space [3]. So, the memory access time depends on the location of the accessed data.

Multithreading can also achieve higher instruction rates on processors which contain multiple functional units (e.g. superscalars) or multiple-processing elements (e.g. chip multiprocessors) [4]. To achieve higher performance, it is therefore necessary to optimize the number of synchronization and execution units (or) to fin an appropriate multi-threaded model.

In [5], each thread is comprised of conventional control-fl w instructions. These instructions do not retain functional properties and need a Write-after-Write (WAW) and Write-After-Read (WAR) dependencies [6].

In [7], a few limitations of the pure datafl w model are presented. They are: 1) too-fin grained (instruction level) multithreading and 2) difficult in exploiting the memory hierarchies and registers [7]. However, in the model developed in [8], the instructions within a thread retain functional properties of datafl w model and thus eliminates the need for complex hardware. Our work models the datafl w instructions appearing in [8].

Section II shows the block diagram of the closed queuing network model with multiple servers, and describes the model in detail. Section III discusses the performance measures related to the system model with multiple servers. Section IV discusses the simulation results of the multiple-server system model. Finally, Section V presents the conclusions.

II. System Model

Figure 1 represents a good approximation to an open-queuing network model having multiple servers (for each queue) under heavy-load conditions.

Each job circulating in the closed network is said to be an active job and must be allocated a partition of main memory. The number of active jobs, k_i, at server i among the SPs, is the number of jobs (tasks) currently being served by server i. The number of active jobs, k'_j, at server j among the EPs, is the number of jobs (tasks) currently being served by server j. For a closed queuing model with m servers for SPs and n servers for EPs, N is the total number of tasks currently being

[1]**Corresponding Author**: Member of Departement Genie des Systemes d'Information et de Telecommunications (GSIT).

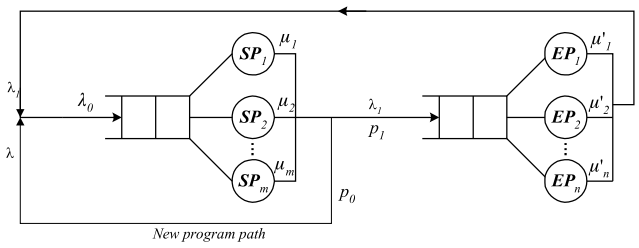

Fig. 1. **Multiple servers - Closed queuing model with feedback**

served by servers i ($1 \leq i \leq m$) and servers j ($1 \leq j \leq n$). Or, $N = k_1 + k_2 + \ldots + k_m + k'_1 + k'_2 + \ldots + k'_n$, where $N \geq (m+n)$ is the total number of jobs in the system. The total number of active jobs, N, is called the degree of multi-programming [1].

The number of jobs in the system is restricted to N due to the constraints of finit main memory [9]. Ready jobs will wait at their terminals until some job leaves from the active set, at which time one of the ready jobs enters the active set and is allowed to compete for the system resources [9].

Let the external arrival rate be λ, arrival rate to all SPs be λ_0, and the arrival rate to the EPs be λ_1. So,

$$\lambda_1 = \lambda_0 p_1. \tag{1}$$

The total arrival rate to all SPs is equal to sum of the external arrival rate and the arrival rate to all EPs. Hence,

$$\lambda_0 = \lambda + \lambda_1. \tag{2}$$

Substituting Equation 1 in Equation 2 and rearranging,

$$\lambda_0 = \frac{\lambda}{1 - p_1}. \tag{3}$$

Also, we have

$$p_0 + p_1 = 1. \tag{4}$$

Substituting Equation 4 in Equation 3, we have

$$\lambda_0 = \frac{\lambda}{p_0}. \tag{5}$$

The arrival rate to the EPs is

$$\begin{aligned} \lambda_1 &= \lambda_0 p_1 \\ &= \frac{\lambda}{p_0} p_1. \end{aligned} \tag{6}$$

Let μ_i be the service rate (rate at which tasks are being served) of server i among the SPs, and let μ'_j be the service rate of server j among the EPs. The utilization of each SP_i ($1 \leq i \leq m$) is

$$\begin{aligned} \rho_i &= \frac{\lambda_0}{\mu_i} \\ &= \frac{\lambda}{p_0 \mu_i}, \end{aligned} \tag{7}$$

and the utilization of each EP_j ($1 \leq j \leq n$) is

$$\begin{aligned} \rho'_j &= \frac{\lambda_1}{\mu'_j} \\ &= \frac{\lambda p_1}{p_0 \mu'_j}. \end{aligned} \tag{8}$$

Consider the closed network of queues shown in Figure 1. The state of the network is given by an (m+n)-tuple, $s = (k_1, k_2, \ldots, k_m, k'_1, k'_2, \ldots, k'_n)$.

Assuming that the service times of all servers are exponentially distributed, the stochastic process modelling the behavior of the network is a finite-stat homogeneous continuous-time Markov chain (CTMC), which can be shown to be irreducible and recurrent non-null [1] (assuming that $0 < p_0 \leq 1$ and $0 < p_1 \leq 1$).

The transient probability matrix, $\mathbf{T}$, of the Discrete-time Markov chain (DTMC) is given by

$$\mathbf{T} = \begin{bmatrix} p_0 & p_1 \\ 1 & 0 \end{bmatrix}. \tag{9}$$

The DTMC is finite and if we assume that $0 < p_0 \leq 1$ and $0 < p_1 \leq 1$, then the DTMC can be shown to be irreducible and periodic. Then, the unique relative visit count vector $\underline{\mathbf{v}} = (v_0, v_1)$ can be obtained by solving the system of equations,

$$\underline{\mathbf{v}} = \underline{\mathbf{v}}\mathbf{T}. \tag{10}$$

If we observe the system for a real-time interval of duration τ, then $v_i \tau$ can be interpreted to be the average number of visits to node i in the interval [1].

The terms $v_0\tau$ and $v_1\tau$ represent the average number of visits to SPs and EPs respectively. In this sense, v_0 can be thought of as a relative visit count in the SPs (or the relative arrival rate to the SPs through the new program path), and v_1 as a relative visit count in the EPs. Thus, v_0 (and v_1) represent the relative throughputs of the respective nodes.

For the network of queues shown in Figure 1, Equation 10 becomes

$$[v_0 \; v_1] = [v_0 \; v_1]\mathbf{T}. \tag{11}$$

The system of linear equations represented by Equation 11 is

$$\begin{aligned} v_0 &= v_0 p_0 + v_1 \Rightarrow v_1 = v_0(1 - p_0) \\ v_1 &= v_0 p_1. \end{aligned} \tag{12}$$

It is clear that both the equations shown above are identical because $p_0 + p_1 = 1$. v_0 can be chosen as any real value that will aid us in our computations. The usual choices for v_0 are $\frac{1}{p_0}$, μ_1, and 1. If we choose $v_0 = \frac{1}{p_0}$, then from Equation 12, we have

$$v_1 = \left(\frac{1 - p_0}{p_0}\right). \tag{13}$$

The relative utilization of device i is (SPs only)

$$\rho_i = \frac{v_0}{\mu_i} = \frac{1}{p_0 \mu_i} \tag{14}$$

$\forall\ i = 1,2,\ldots,m$. The relative utilization of device j is (EPs only)

$$\rho'_j = \frac{v_1}{\mu'_j} = \frac{p_1}{p_0 \mu'_j} \tag{15}$$

$\forall\ j = 1,2,\ldots,n$.
Substituting ρ_i and ρ'_j from Equations 14 and 15 in

$$p(k_1,k_2,\ldots,k_m,k'_1,k'_2,\ldots,k'_n) = \frac{1}{C(N)} \prod_{i=1}^{m} \rho_i^{k_i} \prod_{j=1}^{n} (\rho'_j)^{k'_j}, \tag{16}$$

we have

$$p(k_1,k_2,\ldots,k_m,k'_1,k'_2,\ldots,k'_n) = \frac{1}{C(N)} \prod_{i=1}^{m} \left(\frac{1}{p_0 \mu_i}\right)^{k_i} \prod_{j=1}^{n} \left(\frac{p_1}{p_0 \mu'_j}\right)^{k'_j}. \tag{17}$$

The normalization constant can be expressed as

$$C(N) = \sum_{s \in I} \prod_{i=1}^{m} \rho_i^{k_i} \prod_{j=1}^{n} (\rho'_j)^{k'_j}. \tag{18}$$

where $I = \{(k_1,k_2,\ldots,k_m,k'_1,k'_2,\ldots,k'_n)\}$ and s is as define earlier.

A. Normalization constant: Recursive algorithm

The computation of the normalization constant from Equation 18 is very expensive and numerically unstable because the number of states grow exponentially with the number of customers and the number of service centers [1]. It is therefore necessary to derive computationally stable and efficien algorithms to obtain the normalization constant, $C(N)$.
Let us consider the following polynomial in z.

$$\begin{aligned} G(z) &= \prod_{i=1}^{m} \frac{1}{1-\rho_i z} \prod_{j=1}^{n} \frac{1}{1-\rho'_j z} \\ &= \left(1+\rho_1 z+(\rho_1 z)^2+\ldots\right) \\ &\quad \times \left(1+\rho_2 z+(\rho_2 z)^2+\ldots\right)\ldots \\ &\quad \times \left(1+\rho_m z+(\rho_m z)^2+\ldots\right) \\ &\quad \times \left(1+\rho'_1 z+(\rho'_1 z)^2+\ldots\right) \\ &\quad \times \left(1+\rho'_2 z+(\rho'_2 z)^2+\ldots\right)\ldots \\ &\quad \times \left(1+\rho'_n z+(\rho'_n z)^2+\ldots\right). \end{aligned} \tag{19}$$

Now, the coefficien of $G(z)$ is equal to the normalization constant, $C(N)$, since the coefficien is the sum of all terms of the form $\rho_1^{k_1}\rho_2^{k_2}\ldots\rho_m^{k_m}(\rho'_1)^{k'_1}(\rho'_2)^{k'_2}\ldots(\rho'_n)^{k'_n}$ with $\sum_{i=1}^{m} k_i + \sum_{j=1}^{n} k'_j = N$. In other words, $G(z)$ is the generating function of the sequence $C(1), C(2), \ldots$
We can write

$$\begin{aligned} G(z) &= \sum_{N=0}^{\infty} C(N) z^N \\ &= C(0) + C(1)z + C(2)z^2 + \ldots \end{aligned} \tag{20}$$

where $C(0)$ is define to be equal to unity [1]. Since $C(N)$ is not a probability, $G(z)$ is not necessarily equal to unity. The polynomial $G(z) \geq 0$. In order to derive a recursive relation for computing $C(N)$, we defin

$$\begin{aligned} G_{i,j}(z) &= \prod_{k=1}^{i} \frac{1}{(1-\rho_k z)} \prod_{l=1}^{j} \frac{1}{(1-\rho'_l z)} \\ &= \frac{1}{(1-\rho_1 z)(1-\rho_2 z)\ldots(1-\rho_i z)} \\ &\quad \times \frac{1}{(1-\rho'_1 z)(1-\rho'_2 z)\ldots\left(1-\rho'_j z\right)}, \end{aligned} \tag{21}$$

where $i = 1,2,\ldots,m$ and $j = 1,2,\ldots,n$. Here, $G_{m,n}(z) = G(z)$. Also, let us defin $C_{i,j}(l)$ as

$$G_{i,j}(z) = \sum_{l=0}^{\infty} C_{i,j}(l) z^l \tag{22}$$

where $i = 1,2,\ldots,m$ and $j = 1,2,\ldots,n$ so that $C_{m,n}(l) = C(l)$. Let

$$G_{1,0}(z) = \frac{1}{1-\rho_1 z}. \tag{23}$$

The generalized expression is

$$G_{i,j}(z) = G_{i-1,j}(z) \frac{1}{1-\rho_i z}. \tag{24}$$

Now, Equation 24 can be rewritten as

$$\begin{aligned} G_{i,j}(z)(1-\rho_i z) &= G_{i-1,j}(z) \\ \text{i.e.,}\quad G_{i,j}(z) &= \rho_i z G_{i,j}(z) + G_{i-1,j}(z) \\ \text{i.e.,}\quad \sum_{l=0}^{\infty} C_{i,j}(l) z^l &= \sum_{l=0}^{\infty} \rho_i z C_{i,j}(l) z^l + \sum_{l=0}^{\infty} C_{i-1,j}(l) z^l. \end{aligned} \tag{25}$$

Equating the coefficient of z^l on both sides, we have a recursive formula,

$$C_{i,j}(l) = \rho_i C_{i,j}(l-1) + C_{i-1,j}(l), \tag{26}$$

where $i = 1,2,\ldots,m$, $j = 1,2,\ldots,n$ and $l = 1,2,\ldots,N$. The initialization is obtained from Equations 22 and 23 as

$$C_{1,0}(l) = \rho_1^l \ \forall\ l = 0,1,\ldots,N. \tag{27}$$

Also from Equation 21, we have the coefficien of z^0 in $G_{i,j}(z)$ as unity. Hence,

$$C_{i,j}(0) = 1 \quad \forall\, i = 1,2,\ldots,m \text{ and } \forall\, j = 1,2,\ldots,n. \tag{28}$$

B. Utilization of i^{th} device

Consider a slight modificatio to the generating function, $G(z)$, denoted as $H_i(z)$:

$$\begin{aligned} H_i(z) &= \left(\prod_{j=1(j\neq i)}^{m} \frac{1}{1-\rho_j z}\right)\left(\prod_{k=1}^{n} \frac{1}{1-\rho'_k z}\right) \\ &\times \left(\frac{1}{1-\rho_i z} - 1\right) \\ &= \left(1+\rho_1 z+(\rho_1 z)^2+\ldots\right)\ldots \\ &\times \left(1+\rho_{i-1} z+(\rho_{i-1} z)^2+\ldots\right) \\ &\times \left(\rho_i z+(\rho_i z)^2+\ldots\right) \\ &\times \left(1+\rho_{i+1} z+(\rho_{i+1} z)^2+\ldots\right)\ldots \\ &\times \left(1+\rho_m z+(\rho_m z)^2+\ldots\right) \\ &\times \left(1+\rho'_1 z+(\rho'_1 z)^2+\ldots\right)\ldots \\ &\times \left(1+\rho'_n z+(\rho'_n z)^2+\ldots\right). \end{aligned} \tag{29}$$

The difference between $H_i(z)$ and $G(z)$ is that the firs term is omitted in the factor corresponding to the i^{th} device. So, the coefficien of z^N in $H(z)$ will be the sum of all terms $\rho_1^{k_1}\rho_2^{k_2}\ldots\rho_m^{k_m}(\rho'_1)^{k'_1}(\rho'_2)^{k'_2}\ldots(\rho'_n)^{k'_n}$, where $k_i \geq 1\ \forall\ i = 1,2,\ldots,m$ and $k'_j \geq 1\ \forall\ j = 1,2,\ldots,n$. Here, ρ_i is the relative utilization of device i ($1 \leq i \leq m$), and ρ'_j is the relative utilization of device j ($1 \leq j \leq n$).

¿From Equation 17, we see that the coefficien of z^N in $G(z)$ yields the marginal probability $P(N_i \geq 1)$, which is exactly the utilization $U_i(N)$. So,

$$\begin{aligned} H_i(z) &= \frac{G(z)\left[\frac{1}{1-\rho_i z} - 1\right]}{\frac{1}{1-\rho_i z}} \\ &= \rho_i z G(z). \end{aligned} \tag{30}$$

From Equation 30, the coefficien of z^N in $H_i(z)$ is simply ρ_i times the coefficien of z^{N-1} in $G(z)$. Therefore, we have

$$\begin{aligned} U_i(N) &= \rho_i \frac{C(N-1)}{C(N)} \\ &= \left(\frac{1}{p_0\mu_i}\right)\frac{C(N-1)}{C(N)} \end{aligned} \tag{31}$$

as the utilization of the i^{th} device among the SPs.

Similarly, the utilization of the j^{th} device among the EPs is

$$\begin{aligned} U'_j(N) &= \rho'_j \frac{C(N-1)}{C(N)} \\ &= \left(\frac{p_1}{p_0\mu'_j}\right)\frac{C(N-1)}{C(N)}. \end{aligned} \tag{32}$$

C. Relative Utilizations

The relative utilizations of SPs and EPs when there are N jobs in the system are given by $U_i(N)/U'_j(N)$. From Equations 31 and 32, the relative utilizations between SPs and EPs is given by

$$\frac{U_i(N)}{U'_j(N)} = \frac{\rho_i}{\rho'_j}. \tag{33}$$

Substituting Equations 14 and 15 into Equation 33, we have

$$\frac{U_i(N)}{U_j(N)} = \left(\frac{\mu'_j}{\mu_i}\right)\left(\frac{1}{p_1}\right). \tag{34}$$

Equation 34 explains the reason for calling ρ_i and ρ'_j as "relative utilizations".

III. Performance Measures

A. Queue Lengths

The probability that there are k or more jobs at node i is given by [1]

$$P(N_i \geq k) = \rho_i^k \frac{C(N-k)}{C(N)}. \tag{35}$$

The average queue length at node i in the SPs when there are N jobs in the system is given by

$$E[L_i(N)] = \sum_{j=1}^{N} \rho_i^j \frac{C(N-j)}{C(N)}. \tag{36}$$

Similarly, the average queue length at node j in the EPs when there are N jobs in the system is given by

$$E\left[L'_j(N)\right] = \sum_{l=1}^{N} (\rho'_j)^l \frac{C(N-l)}{C(N)}. \tag{37}$$

The average number of jobs in the system is equal to the sum of the average number of jobs in SPs and EPs. i.e.,

$$\begin{aligned} E[L] &= E[L_i(N)] + E\left[L'_j(N)\right] \\ &= \sum_{j=1}^{N} \rho_i^j \frac{C(N-j)}{C(N)} + \sum_{l=1}^{N} (\rho'_j)^l \frac{C(N-l)}{C(N)} \\ &= \sum_{j=1}^{N} \frac{\left(\frac{1}{p_0\mu_i}\right)^j C(N-j)}{C(N)} + \sum_{l=1}^{N} \frac{\left(\frac{p_1}{\mu'_j p_0}\right)^l C(N-l)}{C(N)} \end{aligned} \tag{38}$$

where $i = 1,2,\ldots,m$ and $j = 1,2,\ldots,n$.

B. Normalization Constant

¿From Equation 38, the total number of jobs in the closed queuing network is given by

$$\begin{aligned} N &= \sum_{i=1}^{m} E[L_i(N)] + \sum_{j=1}^{n} E\left[L'_j(N)\right] \\ &= \frac{1}{C(N)}\left[\sum_{i=1}^{m}\sum_{j=1}^{N} \rho_i^j C(N-j)\right. \\ &+ \left.\sum_{j=1}^{n}\sum_{l=1}^{N} \left(\rho'_j\right)^l C(N-l)\right], \end{aligned} \tag{39}$$

where $\rho_i = \frac{1}{p_0\mu_i}$ and $\rho'_j = \frac{p_1}{p_0\mu'_j}$.
Cross-multiplying in Equation 39, an alternative recursive formula for the computation of $C(N)$ is

$$C(N) = \frac{1}{N}\sum_{l=1}^{N} C(N-l)\left[\sum_{i=1}^{m}\rho_i^l + \sum_{j=1}^{n}\left(\rho'_j\right)^l\right] \tag{40}$$

with the initial condition $C(0) = 1$. Equation 40 requires more arithmetic operations than Equation 26 to compute the normalization constant. But, it is more efficien to pre-compute the factors $\sum_{i=1}^{m}\rho_i^j + \sum_{l=1}^{n}\left(\rho'_l\right)^j$ for each i and j, and uses the non-recursive formula. For the given closed queuing network, the total estimated number of jobs in the system is $\sum_{i=1}^{m} E[L_i(N)] + \sum_{j=1}^{n} E\left[L'_j(N)\right]$.

C. Response times

The average response time of node i in SPs ($1 \le i \le m$) is given by

$$\begin{aligned} R_{SP_i} &= \frac{E[L_i(N)]}{\lambda_0} \\ &= \frac{1}{\lambda_0}\sum_{j=1}^{N}\frac{\rho_i^j C(N-j)}{C(N)} \\ &= \frac{p_0}{\lambda}\sum_{j=1}^{N}\frac{\left(\frac{1}{p_0\mu_i}\right)^j C(N-j)}{C(N)}, \end{aligned} \tag{41}$$

since $\lambda_0 = \frac{\lambda}{p_0}$.

The average response time of node j in EPs ($1 \le j \le n$) is given by

$$\begin{aligned} R_{EP_j} &= \frac{E\left[L'_j(N)\right]}{\lambda_1} \\ &= \frac{1}{\lambda_0 p_1}\sum_{l=1}^{N}\frac{\left(\rho'_j\right)^l C(N-l)}{C(N)} \\ &= \frac{p_0}{\lambda p_1}\sum_{l=1}^{N}\frac{\left(\frac{p_1}{p_0\mu'_j}\right)^l C(N-l)}{C(N)}, \end{aligned} \tag{42}$$

since $\lambda_1 = \lambda_0 p_1$.
The total response time of the system (SPs and EPs) is

$$\begin{aligned} R &= R_{SP_i} + R_{EP_j} \\ &= \frac{p_0}{\lambda}\left\{\sum_{j=1}^{N}\rho_i^j\frac{C(N-j)}{C(N)} + \sum_{l=1}^{N}\left(\rho'_j\right)^l\frac{C(N-l)}{C(N)}\right\}, \end{aligned} \tag{43}$$

where $\rho_i = \frac{1}{p_0\mu_i}$ and $\rho'_j = \frac{p_1}{p_0\mu'_j}$.

D. Average Waiting times

The average waiting time at queue Q_{SP} is given by

$$\begin{aligned} E[W] &= R_{SP} - \frac{1}{\mu} \\ &= \frac{p_0}{\lambda C(N)}\sum_{j=1}^{N}\left(\frac{1}{p_0\mu}\right)^j C(N-j) - \frac{1}{\mu}. \end{aligned} \tag{44}$$

Here, $\frac{1}{\mu}$ is the average service time of servers in the SPs. The average waiting time at queue Q_{EP} is given by

$$\begin{aligned} E\left[W'\right] &= R_{EP} - \frac{1}{\mu'} \\ &= \frac{p_0}{\lambda p_1 C(N)}\sum_{l=1}^{N}\left(\frac{p_1}{p_0\mu'}\right)^l C(N-l) - \frac{1}{\mu'}. \end{aligned} \tag{45}$$

Here, $\frac{1}{\mu'}$ is the average service time of servers in the EPs. The average waiting time in the system (SPs and EPs), $E[W_s]$, is given by

$$\begin{aligned} E[W_s] &= \frac{p_0}{\lambda}\sum_{j=1}^{N}\frac{\rho^j C(N-j)}{C(N)} - \frac{1}{\mu} \\ &+ \frac{p_0}{\lambda p_1}\sum_{l=1}^{N}\frac{(\rho')^l C(N-l)}{C(N)} - \frac{1}{\mu'} \end{aligned} \tag{46}$$

where $\rho = \frac{1}{p_0\mu}$ and $\rho' = \frac{p_1}{p_0\mu'}$.

E. Utilizations

The steady-state probability of having all jobs served by the EPs is given by

$$\begin{aligned} p(0,0,\ldots,0,k'_1,k'_2,\ldots,k'_n) &= \frac{1}{C(N)}\prod_{j=1}^{n}\left(\rho'_j\right)^{k'_j} \\ &= \frac{1}{C(N)}\prod_{j=1}^{n}\left(\frac{p_1}{p_0\mu'_j}\right)^{k'_j}. \end{aligned} \tag{47}$$

The steady-state probability of having all jobs served by the SPs is given by

$$
\begin{aligned}
U_0 &= p(k_1, k_2, \ldots, k_m, 0, \ldots, 0) \\
&= 1 - p\left(0, \ldots, 0, k'_1, k'_2, \ldots, k'_n\right) \\
&= 1 - \frac{1}{C(N)} \prod_{j=1}^{n} \left(\rho'_j\right)^{k'_j} \\
&= 1 - \frac{1}{C(N)} \prod_{j=1}^{n} \left(\frac{p_1}{p_0 \mu'_j}\right)^{k'_j}. \qquad (48)
\end{aligned}
$$

If $U_0 > 1 - U_0$, or equivalently, $U_0 > \frac{1}{2}$, we have more SP utilization. This indicates that the execution of the program is dominated by SPs. If $U_0 < \frac{1}{2}$, we have more EP utilization. In this case, the execution of the program is dominated by the EPs. When $U_0 = \frac{1}{2}$, the program execution is said to be balanced.

F. Average system throughput

The real utilization of the i^{th} node in the SPs ($1 \leq i \leq m$) is given by

$$U_i(N) = \rho_i \frac{C(N-1)}{C(N)}, \qquad (49)$$

and the real utilization of the j^{th} node in the EPs ($1 \leq j \leq n$) is given by

$$U'_j(N) = \rho'_j \frac{C(N-1)}{C(N)}. \qquad (50)$$

The average throughput of the i^{th} node in the SPs ($1 \leq i \leq m$) is given by

$$
\begin{aligned}
E[T_i(N)] &= \mu_i p_0 U_i(N) \\
&= \mu_i p_0 \rho_i \frac{C(N-1)}{C(N)}. \qquad (51)
\end{aligned}
$$

Now, the system throughput is provided only by the contribution of SPs [1].
The average system throughput is given by

$$
\begin{aligned}
E[T(N)] &= \sum_{i=1}^{m} E[T_i(N)] \\
&= \sum_{i=1}^{m} \mu_i p_0 \rho_i \frac{C(N-1)}{C(N)} \\
&= \sum_{i=1}^{m} \frac{C(N-1)}{C(N)}. \qquad (52)
\end{aligned}
$$

IV. Simulation Results

A simulation is performed for the closed queuing network model with multiple servers. The number of synchronization processors is 5 and the number of execution processors is chosen to be 6. The service rates μ_i and μ'_j, and the probability of jobs getting serviced at Q_{EP}, p_1, are appropriately chosen to have a constant probability of entering the new program path, $p_0 = 0.4$.

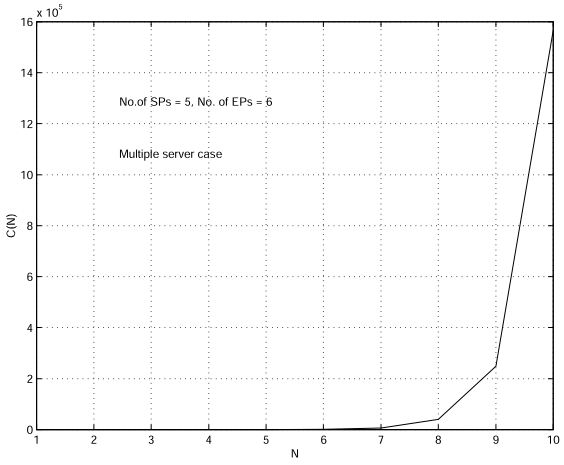

Fig. 2. Normalization constant of a closed queuing network versus the total number of jobs

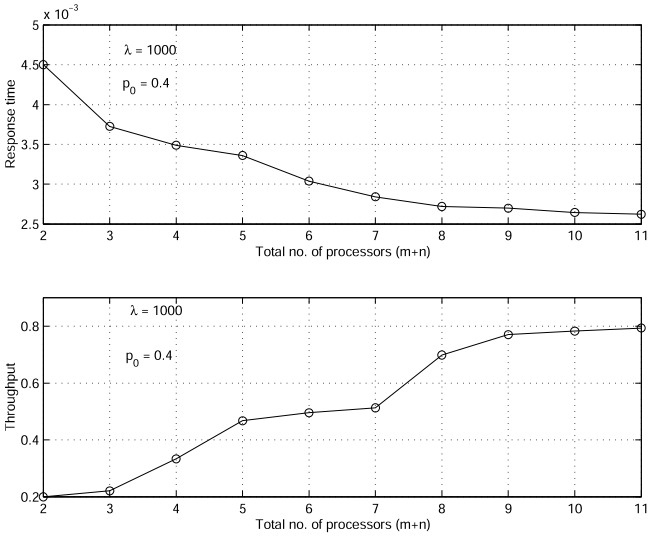

Fig. 3. Response time and throughput versus the total number of processors (SPs and EPs)

TABLE I
TOTAL NUMBER OF JOBS IN SPS, EPS, AND SYSTEM

(m,n)	L_1	L_2	$L = L_1 + L_2$
(1,1)	6.606	2.399	9.006
(1,2)	5.2691	3.0829	8.352
(2,2)	5.8639	2.8571	8.72112
(3,2)	6.42106	2.7643	9.1854
(3,3)	6.24004	3.2577	9.49778
(3,4)	6.12623	3.62211	9.7483
(4,4)	6.6435	3.578	10.2216
(5,4)	7.0656	3.54803	10.6136
(5,5)	7.00727	3.84205	10.8493
(5,6)	6.96239	4.08607	11.04846

The estimated number of jobs in SPs (L_1), EPs (L_2) and jobs in the whole system ($L = L_1 + L_2$), are shown in Table I. It is found that the estimated total number of jobs in the system are close to N, where N is chosen as 10 in the simulation. The real utilizations, $U_i(N)$ and $U'_j(N)$, and the relative utilizations, ρ_i and ρ'_j are tabulated in Tables II and III respectively.

The normalization constant is computed from Equation 40 by pre-computing the utilizations at each of the nodes in SPs and EPs. The normalization constant is plotted against N, the number of jobs in the system in Figure 2. Knowing the normalization constant, the total system response time is computed from Equation 43. The response time is plotted in Figure 3. For appropriate choices of the service rates of SPs and EPs, the response time is found to decrease as the total number of processors increases.

The average system throughput is obtained from Equation 52 and is plotted in Figure 3 against the total number of processors, (m+n). The system throughput increases as the total number of processors increases. The total number of arrivals, $\lambda = 1000$ and $p_0 = 0.4$ are kept constant throughout the simulation.

V. Conclusions

In this paper, we introduced a closed network of queues to model datafl w in a multi-processor system. The instruction streams are executed simultaneously (multi-threading) to minimize the loss of CPU cycles. A convolution algorithm is used to compute the normalization constant as a function of the degree of multiprogramming (number of active jobs) in the queuing model. The relative utilizations and other system performance measures are also derived for the multiple-server network model. The response time and throughput are plotted against the total number of processors in the system model.

References

[1] K. Trivedi, *Probability & Statistics with reliability, queuing and computer science applications.* New Jersey: Prentice-Hall, 1982.

[2] B. Shankar, L. Rho, W. Bohm, and W. Najjar, "Control of parallelism in multithreaded code," *Proc. of the Intl Conference on Parallel Architectures and Compiler Techniques (PACT-95)*, June 1995.

[3] W. Grunewald and T. Ungerer, "A multithreaded processor design for Distributed Shared Memory (DSM) system," *Proc. of the Intl Conference on Advances in parallel and distributed computing*, 1997.

[4] M. Lam and R. Wilson, "Limits of control fl w on parallelism," *Proc. of the 19th Intl Symposium on Computer Architecture (ISCA-19)*, pp. 46–57, May 1992.

[5] S. Sakai, "Architectural and software mechanisms for optimizing parallel computations," *Proc. of 1993 Intl Conference on Supercomputing*, July 1993.

[6] K. M. Kavi, J. Arul, and R. Giorgi, "Execution and cache performance of the scheduled datafl w architecture," *Journal of Universal Computer Science*, Oct. 2000.

[7] M. Takesue, "A unifie resource management and execution control mechanism for datafl w machines," *Proc. 14th Int'l Symp. on Computer Architecture (ICSA-14)*, pp. 90–97, June 1987.

[8] K. M. Kavi, R. Girogi, and J. Arul, "Scheduled datafl w: Execution paradigm, architecture, and performance evaluation," *IEEE Transactions on Computers*, vol. 50, no. 8, pp. 834–846, Aug. 2001.

[9] L. Kleinrock, *Queuing systems, Volume II: Computer Applications.* John Wiley & Sons Inc., 1976.

TABLE II
REAL UTILIZATIONS OF SPS AND EPS

(m,n)	$U_i(N),\ i=1,2,\ldots,m$	$U'_j(N),\ j=1,2,\ldots,n$
(1,1)	(0.625)	(0.375)
(1,2)	(0.61349)	(0.22085, 0.1656)
(2,2)	(0.4167, 0.2083)	(0.25, 0.125)
(3,2)	(0.3246, 0.1948, 0.12987)	(0.2337, 0.1168)
(3,3)	(0.2751, 0.15287, 0.1179)	(0.24765, 0.1238, 0.08255)
(3,4)	(0.2137, 0.14246, 0.10958)	(0.25643, 0.1282, 0.08547, 0.0641)
(4,4)	(0.1454, 0.1247, 0.0969, 0.0872)	(0.26186, 0.1309, 0.0872, 0.0654)
(5,4)	(0.3851, 0.19255, 0.077, 0.035, 0.077)	(0.1155, 0.04621, 0.0385, 0.033)
(5,5)	(0.3912, 0.1956, 0.04891, 0.04891, 0.04891)	(0.11738, 0.04695, 0.03912, 0.0335, 0.02934)
(5,6)	(0.092267, 0.107229, 0.0762, 0.0748, 0.06612)	(0.23804, 0.11902, 0.07934, 0.05951, 0.0476, 0.0396)

TABLE III
RELATIVE UTILIZATIONS OF SPS AND EPS

(m,n)	$\rho_i,\ i=1,2,\ldots,m$	$\rho'_j,\ j=1,2,\ldots,n$
(1,1)	(3.125)	(1.875)
(1,2)	(2.778)	(1, 0.75)
(2,2)	(2.5, 1.25)	(1.5, 0.75)
(3,2)	(2.0833, 1.25, 0.833)	(1.5, 0.75)
(3,3)	(1.667, 0.9259, 0.7142)	(1.5, 0.75, 0.5)
(3,4)	(1.25, 0.833, 0.64102)	(1.5, 0.75, 0.5, 0.375)
(4,4)	(0.833, 0.7142, 0.555, 0.5)	(1.5, 0.75, 0.5, 0.375)
(5,4)	(2.5, 1.25, 0.5, 0.2272, 0.5)	(0.75, 0.3, 0.25, 0.21428)
(5,5)	(2.5, 1.25, 0.3125, 0.3125, 0.3125)	(0.75, 0.3, 0.25, 0.2143, 0.1875)
(5,6)	(0.5813, 0.6756, 0.4807, 0.4716, 0.4167)	(1.5, 0.75, 0.5, 0.375, 0.3, 0.25)