

Analysis of Dynamic Load Balancing Strategies Using a Combination of Stochastic Petri Nets and Queueing Networks

C.R.M.Sundaram

Department of Computational Science
University of Saskatchewan, Saskatoon, Canada

Y.Narahari

Department of Computer Science and Automation
Indian Institute of Science, Bangalore, India

Abstract

This paper is concerned with the analytical evaluation of two well known dynamic load balancing strategies, namely, *shortest queue routing* (SQR) and *shortest expected delay routing* (SEDR). We overcome the limitations of existing analysis methodologies, using a well known hybrid performance model that combines *generalized stochastic Petri nets* and *product form queueing networks*. Our methodology is applicable to both open queueing network and closed queueing network models of load balancing in distributed computing systems. The results show that for homogeneous distributed systems, SQR outperforms all other policies. For heterogeneous systems, SEDR surprisingly performs worse than SQR at low levels of imbalance in loads. However, with increase in imbalance in load, SEDR expectedly performs better than SQR.

1 Introduction

The problem of balancing the load over the entire system so that its overall performance is optimized is an important research problem [20] in Distributed Computing Systems(DCSs). Routing an arriving job to the processor with the shortest queue length(SQR, shortest queue routing) [17] in the case of homogeneous systems and routing to the processor with the shortest expected delay (SEDR, shortest expected delay routing) [4] in the case of heterogeneous systems have been found to be close to optimal heuristic load balancing strategies.

Analysis of the performance of SQR and SEDR policies through simulation will be computationally expensive. It has been reported by Nelson and Philips

in [17] that, when the number of parallel queues is 32, it took several days of computing on an IBM 3090 main frame to get a mere 5 samples. So, in this paper we make an analytical evaluation of SQR and SEDR policies. Analytical evaluation through Markov chains by setting up exact global balance equations will be practically infeasible as the number of equations grow exponentially with increase in the number of parallel queues. Exact analysis through queueing networks is not possible as the individual queues are no longer independent of one another, and the system loses the product form property. Because of the difficulties in making an exact analysis of SQR and SEDR policies, approximate analysis has been attempted by several researchers [5], [8], [10], [11], [12], [13], [14], [15], [17], [22], [23]. In most of the above papers and also in this paper, we model a DCS as a set of parallel servers, each with its own queue. This is the model that is most commonly used in the literature [4], [6], [9], [22].

Kingman [11] and Flatto [14] studied the SQR problem via transform methods and found expressions for the mean number in the system and the occupancy probabilities. The mean number of jobs is expressed as an infinite sum and the sum simplifies only under a heavy traffic assumption. Halfin [13] discusses the two queue version of this problem with homogeneous exponential servers and employing linear programming techniques obtains bounds for the probability distribution of the number of customers in the system and its expected value in equilibrium. Blanc [5] considers a numerical method based on power series expansions and recursions to calculate state probabilities and moments of queue length distributions. Conolly [9] discusses a queueing model fed by a Poisson arrival stream of jobs, and served by two identical exponential servers, each with a queue, with the customers choosing the shortest queue at the instant of their arrival. He proposes a method to calculate state probabilities by means of inversion of tridiagonal matrices. He also shows that there is hardly any difference between the distribution of the sum of the lengths of two parallel queues and the queue length distribution for the M/M/2 queue. Gubner [12] shows that expectation of the difference of the queue lengths for any two queues is uniformly bounded for all loads. Based on the evidences given in [9] and [12], Nelson [17] derives an approximate closed form expression for the mean response time of a multiple queueing system consisting of identical exponential servers fed by a Poisson stream of jobs. Yuan [22] proposes an approximate numerical solution method to determine equilibrium state probabilities of a queueing system consisting of two heterogeneous exponential servers, wherein, an arriving job to the system is routed to the queue with the shortest expected delay.

In spite of a wealth of results available on approximate analysis, we find that these approximations have the following drawbacks:

1. Different techniques are employed to compute different average performance measures such as mean response time [17], and moments of individual queue length distributions [5]. There is no unifying technique to compute all the desirable average performance measures.
2. Most of the approximations [10],[15],[23] proposed so far lead to accurate results only under light traffic or heavy traffic assumptions.

3. Several approximations such as in [5],[10],[13],[15] have been developed for the 2-server case but they are not generalizable to more than 2-servers.
4. Several approximations such as in [13],[17],[23] assume that the arrival stream of jobs constitute a Poisson stream. However, if SQR or SEDR is employed only in a part of the system, the arrivals to this subsystem in general need not be Poisson.
5. Most of the approximations [13],[17],[22] assume that service times of the individual servers are exponentially distributed.

In this paper, we present an efficient, accurate, and general analytical methodology for the evaluation of the performance of SQR and SEDR policies in DCSs. The methodology is a hybrid technique based on a combination of Product Form Queueing Networks (PFQNs) and Generalized Stochastic Petri Nets (GSPNs) [1] proposed by Balbo, Bruell, and Ghanta [3]. Such an integrated technique is necessary because: (a) PFQNs cannot capture these dynamic routings. (b) Though GSPNs can elegantly model these policies, the resulting models are intractable due to state space explosion. Since GSPNs are used to capture dynamic policies, modelling using hybrid technique overcomes drawbacks 1, 2, and 3 mentioned above. Drawback 4 can be overcome by explicitly modelling that part of the subsystem which sends jobs to the subsystem employing SQR or SEDR policies. Drawback 5 can be overcome in a limited sense by allowing Erlang, hypo-exponential, and hyper-exponential distributions. GSPNs are used as a model for only that part of the system employing SQR (or SEDR) policies and hence the method is computationally efficient.

To investigate the efficacy of this approach, we have considered, in this paper, closed central server systems with SQR (or SEDR) routings. Central server models with state dependent routing functions have been considered in the literature [2],[16],[21]. In these papers, queueing networks with state-dependent routing functions based on rational functions of queue lengths (SQR and SEDR do not belong to this category) have been shown to be product form. Even though we have considered closed queueing models, our results are applicable to open queueing models of load balancing in which the maximum number of jobs that can be present in the submodels employing these dynamic routings is finite.

The rest of the paper is organized as follows. In Section 2, we give a brief introduction to the Integrated PFQN-GSPN Models, and then present the integrated PFQN-GSPN methodology for modelling DCSs with SQR and SEDR routings. In Section 3, we present detailed numerical results and make a comparison of SQR and SEDR policies with some simple state-dependent routing functions. The major results of the numerical experimentation are as follows:

- SQR performs the best among all load balancing policies, for homogeneous DCSs.
- In heterogeneous DCSs, SQR surprisingly performs better than SEDR for very low levels of load imbalances, but is expectedly outperformed by SEDR in other conditions.

The results obtained have been found to be remarkably accurate. The models were analyzed using software packages developed by us for PFQN analysis and GSPN analysis.

2 Integrated Models for SQR and SEDR

The integrated queueing network - Petri net (IQP) modelling originated by Balbo, Bruell, and Ghanta [3] is based on the concept of flow-equivalence. The basic idea is to isolate one or more subsystems of a given system and compute the flow-equivalents for these subsystems. The overall model (also called the high level model) of the given system will then have these subsystems represented by the flow-equivalents. The flow-equivalents of these subsystems (also called the submodels) are computed by evaluating the throughput of these subsystems in isolation for all possible populations. The throughputs thus computed together with associated populations constitute the flow-equivalent. The IQP modelling can be of two types : (i) IQP model with GSPN as the high level model and (ii) IQP model with PFQN as the high level model. If the high level model is a PFQN model, then some of its stations will be the flow-equivalent representations (derived using GSPNs) of the submodels that incorporate non-product form features. If the high level model is a GSPN model, then some of its timed transitions will be the flow-equivalent representations (derived using PFQNs) of the submodels that are product form solvable.

Consider a DCS containing a subsystem in which SQR (or SEDR) is employed. Assume that the rest of the DCS contains only product form components. The basic idea behind modelling the above system using IQP models is to compute the flow-equivalent of the SQR (or SEDR) subsystem using a GSPN model. The DCS can then be evaluated as a PFQN, with the SQR (or SEDR) subsystem represented as a state-dependent node derived using the GSPN model of the subsystem. Since it is always possible to obtain an exact representation of the flow-equivalent server, the only source of error in this technique could be due to the interaction between the subsystem and the rest of the DCS.

To illustrate the IQP modelling technique for SQR and SEDR, we consider a central server model of a DCS as shown in Figure 1. It is to be noted that by considering the central server model, we are only simplifying the presentation, and not undermining the general applicability of our results. Also, we stress that our results are applicable not only to closed queueing models but also to those open queueing models, in which the subnetworks with non-product form components have limited populations, such as in [4]. A typical state of the central server network in Figure 1 is given by $\underline{n} = (n_0, n_1, n_2, \dots, n_m)$ where n_i is the number of jobs in the i^{th} node, $i = 0, 1, 2, \dots, m$. In the state \underline{n} , the jobs after getting executed from node 0 are routed to the i^{th} node with probability $q_i(\underline{n})$, $i = 1, 2, \dots, m$ in such a way that the load across the subsystem is balanced. In the following, we mathematically describe the heuristic policies SQR and SEDR. We also describe a few other simple heuristic policies for load balancing. We describe totally three policies A1, A2, and A3 for homogeneous systems, and four policies,

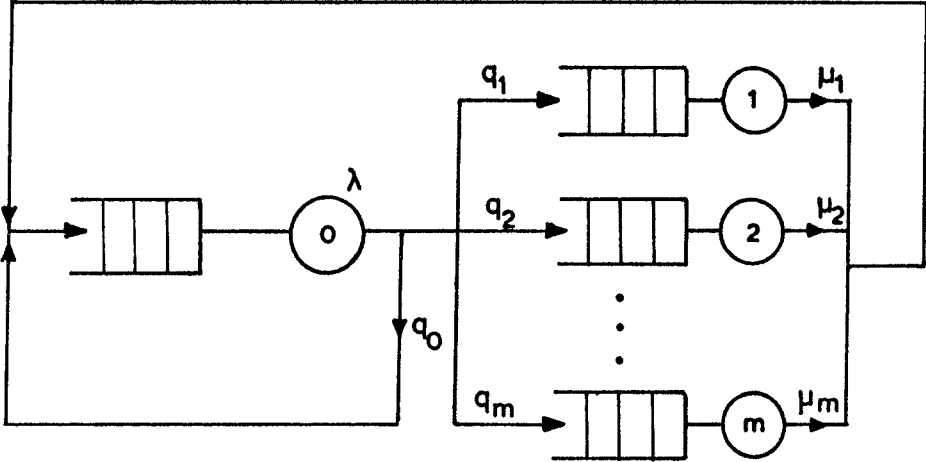


Figure 1: Closed central server model with (m+1) nodes.

B1, B2, B3, and B4, for heterogeneous systems. In the following expressions, μ_i denotes the service rate of the i^{th} node, S denotes the state space of the central server network, \underline{n} is any state in S , and the index i takes the values, $1, 2, \dots, m$.

A. Homogeneous Systems:

$$A1: q_i(\underline{n}) = \frac{1-q_0}{m}$$

$$A2: q_i(\underline{n}) = \frac{(1-q_0) \sum_{j \neq i} n_j}{(m-1) \sum_{j=1}^m n_j}$$

$$A3: q_i(\underline{n}) = \begin{cases} \frac{1-q_0}{|\text{SQR}(\underline{n})|} & \text{if } i \in \text{SQR}(\underline{n}) \\ 0 & \text{otherwise} \end{cases}$$

where $\text{SQR}(\underline{n}) = \{ k: n_k = \min(n_1, n_2, \dots, n_m) \}$

B. Heterogeneous Systems:

$$\text{B1: } q_i(\underline{n}) = \frac{\mu_i(1-q_0)}{\sum_{j=1}^m \mu_j}.$$

$$\text{B2: } q_i(\underline{n}) = \frac{\left(\sum_{j \neq i} \frac{n_j + 1}{\mu_j} \right)^{(1-q_0)}}{(m-1) \sum_{j=1}^m \frac{n_j + 1}{\mu_j}}.$$

B3: same as Policy A3

$$\text{B4: } q_i(\underline{n}) = \begin{cases} \frac{1-q_0}{|\text{SEDR}(\underline{n})|} & \text{if } i \in \text{SEDR}(\underline{n}) \\ 0 & \text{otherwise} \end{cases}$$

$$\text{where } \text{SEDR}(\underline{n}) = \{k: \frac{n_k+1}{\mu_k} = \min(\frac{n_1+1}{\mu_1}, \frac{n_2+1}{\mu_2}, \dots, \frac{n_m+1}{\mu_m})\}$$

The intuition behind policy A1 is that routing with equal probabilities to one of the nodes 1,2,...,m will balance the loads across the subsystem, since all the server speeds are identical. This can be analyzed by product form queueing networks. In policy A2, it is to be observed that, in a given state \underline{n} , the expressions computed are in such a way that, if $n_k \leq n_l$ then $q_k(\underline{n}) \geq q_l(\underline{n})$. So, the shortest queue will have the maximum probability and the longest queue will have the minimum probability, and the queues having queue lengths in between will have probabilities in such a way that the queue having a queue length close to that of the shortest queue will be assigned greater probability than the one having queue length close to that of the longest queue. The central server model employing this policy is non-product form and can be analyzed by GSPNs. The intuition behind policy B1 is that allocating a job to a faster server with greater probability than to a slower server, would minimize the overall expected response time. The central server model employing this policy is product form solvable. Policy B2 is a generalization of policy A2 presented for the homogeneous systems, and again the intuition is that allocating an arriving job with greater probability to the queue having less expected delay would minimize the overall average response time of a job. Central server model employing this policy is not product form solvable, but can be analyzed by GSPNs.

Figure 2 shows the GSPN model of the subsystem with dynamic routing for a closed central server model with 5 nodes ($m = 4$). The reader is referred to [1] for an excellent review of GSPNs. Table 1 gives the interpretation of the places and transitions of the GSPN model. The table also specifies the dynamic random switch in the GSPN model, which models the dynamic routing in the network.

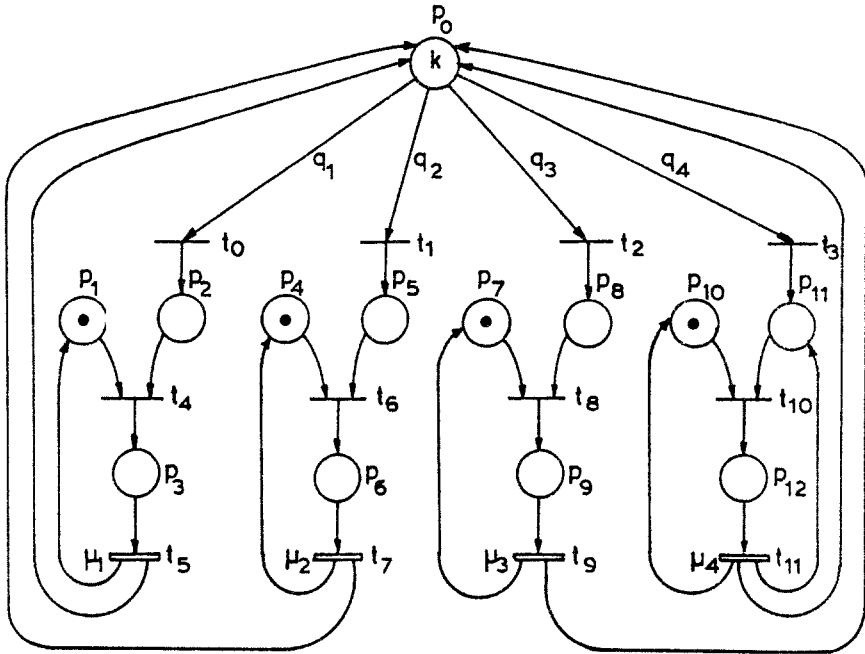


Figure 2: GSPN model for computing flow-equivalents.

Places:

- P_0 : jobs just entered into the subsystem.
 $P_{1+3(i-1)}, i=1,2,3,4$: availability of node i .
 $P_{2+3(i-1)}, i=1,2,3,4$: jobs waiting for node i .
 $P_{3+3(i-1)}, i=1,2,3,4$: jobs getting processed in node i .

Immediate Transitions:

- $t_{0+(i-1)}, i=1,2,3,4$: job is routed to node i after a burst from node 0.
 $t_{4+2(i-1)}, i=1,2,3,4$ jobs start processing in node i

Exponential Transitions:

- $t_{5+2(i-1)}, i=1,2,3,4$: processing of jobs in node i .

Dynamic Random Switch:

- $t_0-q_1; t_1-q_2; t_2-q_3; t_3-q_4$.

These probabilities are computed using the equations in Section 2.

Initial Marking:

- $P_0-k; P_1-1; P_4-1; P_7-1; P_{10}-1$;
-

Table 1: Legend for GSPN Model in Figure 2.

Figure 2 becomes self-explanatory with Table 1. This model captures the subsystem in isolation and is obtained by shorting node 0. The important part of this model is the random switch, comprising the transitions t_1, t_2, t_3 , and t_4 which describes the dynamic routing. The associated probabilities q_1, q_2, q_3 , and q_4 can be defined as in the above equations, in terms of the current marking M of the GSPN model, as given below:

Shortest Queue Routing:

$$q_i(M) = \begin{cases} \frac{1}{|\text{SQRTIE}(M)|} & \text{if } i \in \text{SQRTIE}(M), \quad i=1,2,3, 4 \\ 0 & \text{otherwise} \end{cases}$$

where the set SQRTIE(M) is given by

$$\left\{ i: M(P_{2+3(i-1)}) + M(P_{3+3(i-1)}) = \min_{1 \leq j \leq 4} (M(P_{2+3(j-1)}) + M(P_{3+3(j-1)})) \right\}$$

Shortest Expected Delay Routing:

$$q_i(M) = \begin{cases} \frac{1}{|\text{SEDRTIE}(M)|} & \text{if } i \in \text{SEDRTIE}(M), \quad i=1,2,3, 4 \\ 0 & \text{otherwise} \end{cases}$$

where the set SEDRTIE(M) is given by

$$\left\{ i: \frac{M(P_{2+3(i-1)}) + M(P_{3+3(i-1)}) + 1}{\mu_i} = \min_{1 \leq j \leq 4} \left(\frac{M(P_{2+3(j-1)}) + M(P_{3+3(j-1)}) + 1}{\mu_j} \right) \right\}$$

To compute the flow-equivalent of the subsystem using the GSPN model, one has to evaluate the GSPN model for each possible population. If the population of the closed central server network is N , then the possible populations in the subsystem are $0, 1, 2, \dots, N$. For each population k , the initial marking of the GSPN model can be set by making the number of tokens in place P_0 as k . The sum of the throughput rates of the transitions t_9, t_{10}, t_{11} , and t_{12} will then give the throughput rate of the subsystem. The throughput rates thus computed, together with the associated populations constitute the flow-equivalent of the subsystem.

Figure 3 gives the high level PFQN model of the central server network, with the flow-equivalent of the subsystem constituting just a single node of the PFQN. The PFQN model of Figure 3 can now be solved using standard computational algorithms for PFQNs [7],[18]. It would now be very efficient to study, for example, the effect of variation of the scheduling discipline, service distribution, and the service rate of node 0 on the performance of the overall DCS, since the experimentation is required on a 2-node PFQN. Effectively, we have reduced the evaluation of an overall GSPN model into an evaluation of the GSPN submodel whose state space a cardinality much smaller than that of the overall GSPN model, followed by a 2-node PFQN evaluation.

To see the computational advantage obtained, we show in Table 2, a comparison of the state space size between the GSPN model of Figure 2 (which models

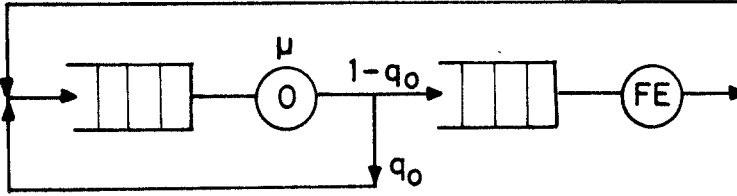


Figure 3: PFQN high level model.

only the subsystem) and the exact GSPN model for the entire system(i.e., the model that captures all the 5 nodes in a GSPN). The significant reduction in the size of the state space is to be noted.

Population	Size of state space of exact GSPN for SQR	Size of state space of subsystem GSPN for SQR	Size of state space of exact GSPN for SEDR	Size of state space of subsystem GSPN for SEDR
4	16	1	12	1
5	48	4	30	2
6	72	6	36	1
7	80	4	48	1
8	81	1	72	1
9	189	4	96	1
10	243	6	120	1
11	255	4	160	1

Table 2: Comparison of state space sizes for detailed and subsystem models.

Generalizing from the example of 5 nodes($m=4$), we get the following result about the cardinality of the state spaces of the GSPN submodel.

Result: For any population, the cardinality of the state space of the GSPN submodel for SQR is less than or equal to $\binom{m}{\frac{m}{2}}$.

Proof: Let the subsystem contain k jobs. Let S be the state space of the GSPN submodel. Take any state $(k_1, k_2, \dots, k_m) \in S$. Since the routing is based on the SQR policy, $|k_i - k_j| \leq 1 \forall 1 \leq i, j \leq m$. If not, jobs can be exchanged between them making the load more balanced. So, each processor must get $\lceil \frac{k}{m} \rceil + x$ jobs where x can be either zero or 1. Since the total number of jobs is equal to k , x will be equal to 1 for exactly $k - \lceil \frac{k}{m} \rceil = k \bmod m$ processors. These $k \bmod m$

processors can be chosen from k processors in $\binom{m}{k \bmod m}$ ways. This value will be maximum when $k \bmod m = \frac{m}{2}$. Since the maximum value $\binom{m}{\frac{m}{2}}$ is independent of the subsystem population k , the above result holds.

The cardinality of the state space of the GSPN submodel for SEDR depends on the service rates of the various nodes and it is difficult to obtain a tighter bound. However, in all the numerical experiments we carried out, we get much less number of states than for SQR.

3 Numerical Results

For the purpose of numerical experiments, we consider the central server model of Figure 1 with $m = 4$. We first present the results for the homogeneous case ($\mu_1 = \mu_2 = \mu_3 = \mu_4$) and then for the heterogeneous case. Note that SQR is same as SEDR for the homogeneous case. We shall study the variation of response time as a function of a parameter that we shall call as the *imbalance in load*, denoted by ρ . ρ is the ratio of the processing capacity of the subsystem ($\mu_1 + \mu_2 + \mu_3 + \mu_4$) to that of node 0 (λ). ρ was chosen as a parameter of interest since the performance of the central server model is crucially affected by this ratio. We present below the important results [19].

3.1 Results for Homogeneous DCS

Here we compare three routing policies A1, A2, and A3. Policy A1 assigns equal probabilities to all the queues. Policy A2 assigns greater probabilities to shorter queues, whereas policy A3 is the SQR policy. Figure 4 shows the percentage difference in response time between policies A1 and A3, while Figure 5 shows the difference in response time between policies A2 and A3. Two populations $N=4$ and $N=6$ are considered.

As the figures show, the difference in performance is appreciable only for small values of ρ . This is due to the following reason: When ρ is low, the processing capacity of the subsystem is much less than that of node 0. In that case, queues are formed in each of the service centers, and the policies that try to utilize this information perform appreciably better than the policies that either do not use this information or use in a limited way. However, when ρ is high, the processing capacity of the subsystem is much higher than that of node 0. In this case, no queues are formed in the service centers of the subsystem, and the nodes are idle, virtually all the time. So, the policies that use queue length information do not produce significant difference in performance from the ones that do not use this information.

As the figures show, for a given ρ , the percentage difference in response time decreases as J increases. This is due to the following reason: The shortest queue routing tries to balance the load across the subsystem and tries to obtain maximum throughput for a given ρ . Due to this, node 0 is utilized to the

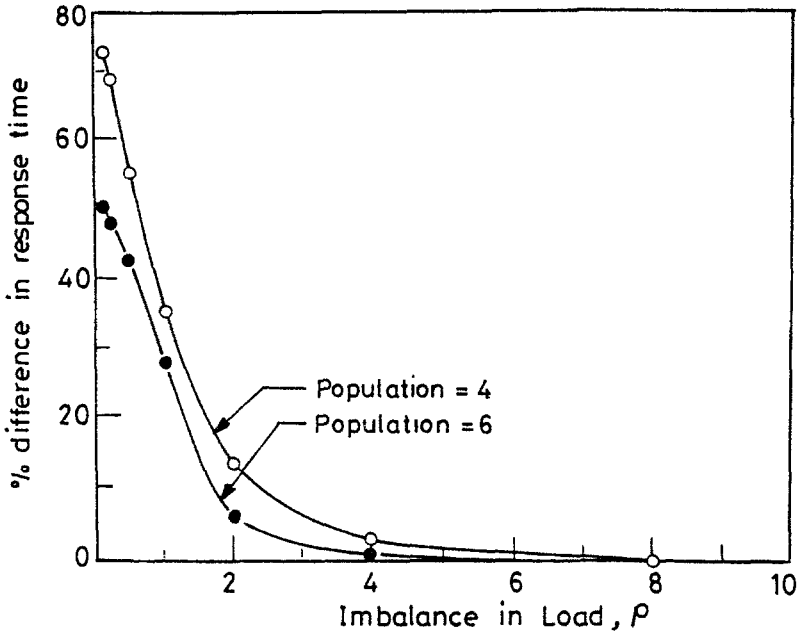


Figure 4: Percentage difference in average response times for Policies A1 and A3.

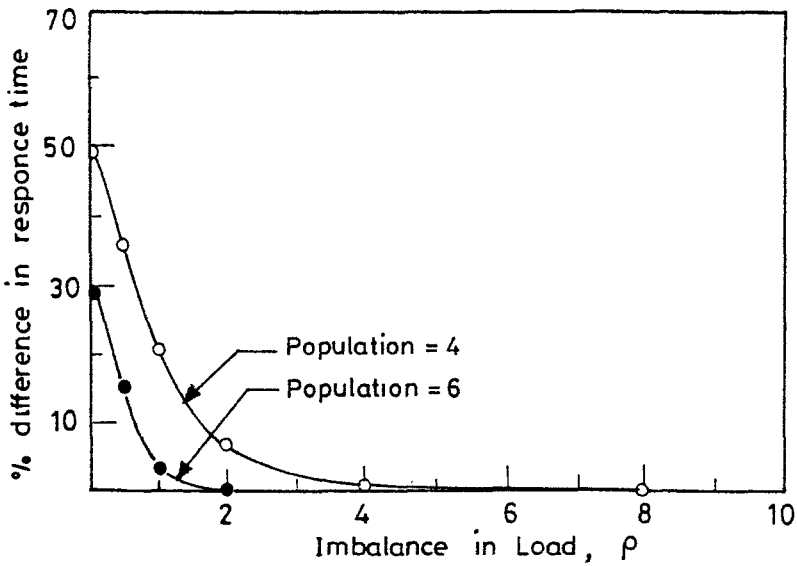


Figure 5: Percentage difference in average response times for Policies A2 and A3.

maximum extent possible for a given ρ . So, increasing the number of jobs from $J=4$ to 6 leads to an increase in response time by 1.5 times. But, in the case of policies A1 and A2, node 0 is not utilized to the maximum extent possible for a given ρ , as the work in the subsystem is not balanced across the nodes. Hence, increasing the number of jobs increases the utilization of node 0 only to a limited extent. Due to this, increase in response time is much less than 1.5 times for $J=6$. So, on the whole, for a given ρ , the percentage difference in response time decreases as J increases. The observations mentioned above can be summarized to the following:

1. SQR outperforms the other two policies for all values of ρ .
2. For a given ρ , the difference in performance between SQR and the other policies will be less for higher populations.
3. The difference in performance between SQR and the other two policies will be very appreciable (as high as 73.05% as observed in the case of policy A1) when ρ is low.

3.2 Results for Heterogeneous DCS

For the purpose of modelling heterogeneous distributed systems, we assumed that service centers in the subsystem have different speeds. Since we cannot exhaust all the possible combinations of service speeds, we consider the following representative cases:

- (i) A system which has two groups of servers, fast and slow (Configuration 1).
- (ii) A system in which each server has a different service rate (Configuration 2).

Here we compare four policies B1, B2, B3, and B4. In policy B1, the job is routed based on the mean service rates of the service centers. Policy B2 uses additional information namely the queue lengths at various service centers apart from the mean service rates. Policies B3 and B4 are SQR and SEDR policies respectively.

Figures 6 and 7 show the comparison between SQR and SEDR for two different configurations. Figure 8 shows the percentage difference in response time between the policies B1 and B2 for the Configuration 1.

As Figure 8 shows, in spite of using additional information, policy B2 performs worse than policy B1. This is due to the following reason: In policy B1, the probability that the job is routed to nodes 1 and 2 is 3 times less than the same for nodes 3 and 4. Once a job is routed, average time spent by a job is 3 times greater than that of a job that goes to nodes 3 and 4. These two factors are complementary and tend to utilize the nodes 1, 2, 3, and 4 more or less the same in policy B1. However, in policy B2, utilizations of nodes 1 and 2 are more than those of nodes 3 and 4. This is because, in all states, the ratio of routing probabilities to nodes 1 and 2 to those of nodes 3 and 4 is not as high as 3 as in

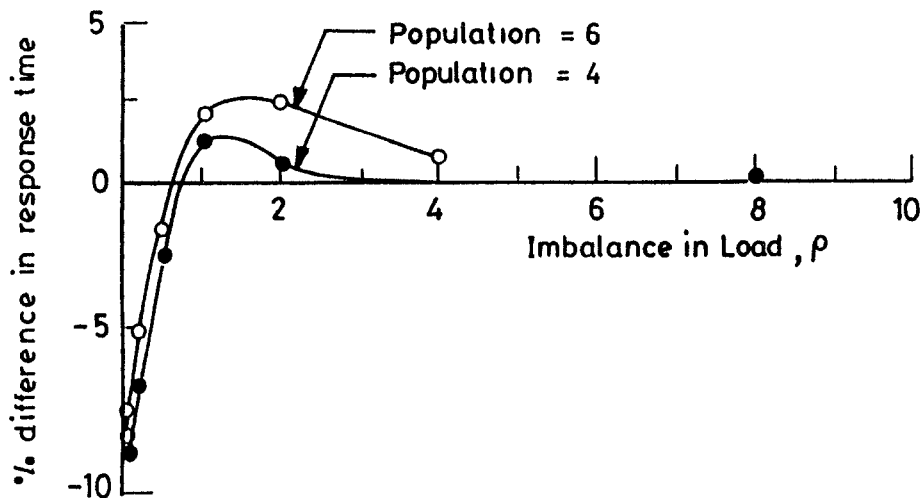


Figure 6: Percentage difference in average response time for policies SQR and SEDR, with $\mu_1 = \mu_2; \mu_3 = \mu_4 = 3\mu_1$.

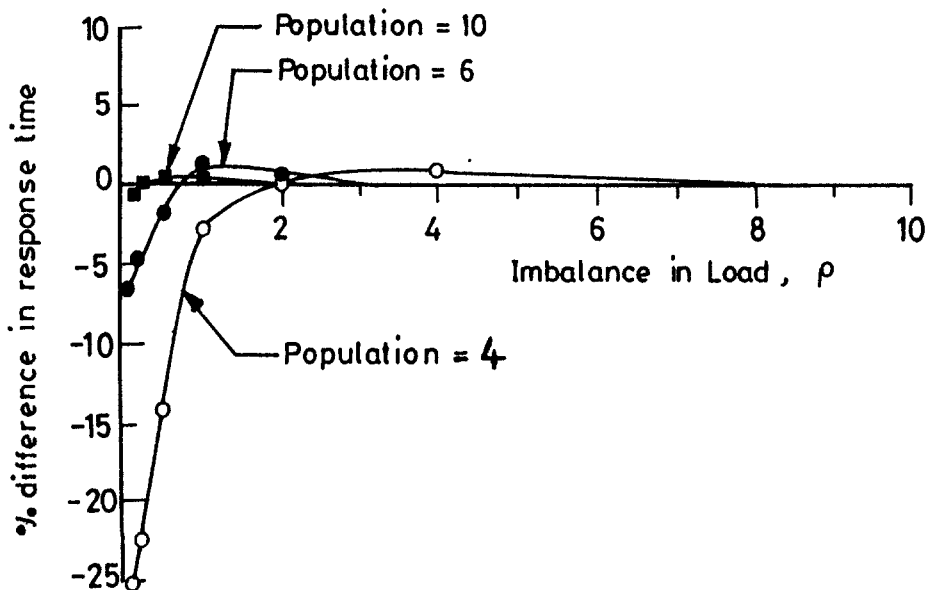


Figure 7: Percentage difference in average response time for policies SQR and SEDR, with $\mu_1 = 2\mu_2; \mu_3 = 3\mu_1; \mu_4 = 4\mu_1$.

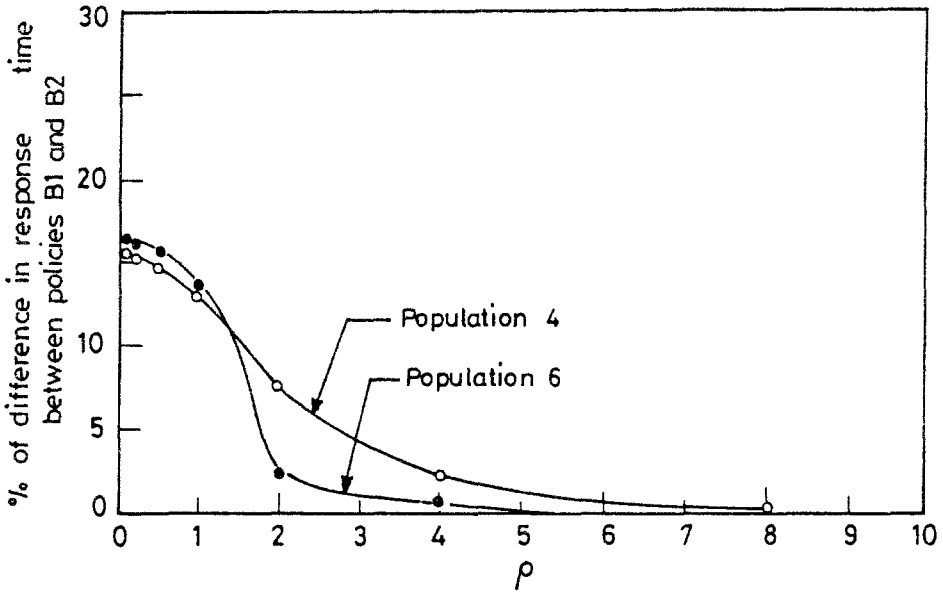


Figure 8: Percentage difference in average response time for policies B2 and B3 for Configuration 1.

policy B1. For example, when the population $J=6$, the ratio of probabilities in the state $n_1 = 3, n_2 = 2, n_3 = 0, n_4 = 0$ is $\frac{22}{23} \approx 1.0$. This makes the throughput of the subsystem much less than that in policy B1, and hence the decrease in response time for policy B2.

As the Figure 6 shows, when ρ is small, SEDR performs worse than the SQR policy. This is due to the following reason: since the service rates of nodes 3 and 4 are 3 times greater than those of nodes 1 and 2, nodes 1 and 2 are not at all utilized in SEDR, where as SQR utilizes all the 4 nodes. So, large queues are formed in nodes 3 and 4 at small values of ρ , and this brings down the performance of SEDR policy below that of SQR. However, as ρ increases, less queues are formed in nodes 3 and 4 due to their higher processing capacity, and the performance of SEDR slowly improves above that of SQR. Similar kind of behaviour is exhibited for the case $J=6$. However, the difference between SQR and SEDR policies at low values of ρ , is not as large as in the case of $J=4$, since nodes 1 and 2 are also utilized $J=6$. As the value of J increases, SEDR will perform close to SQR at low values of ρ .

For configuration 2, we assumed that the service rate of node 2 is twice that of node 1, the service rate of node 3 is thrice that of node 1, and the service rate of node 4 is 4 times that of node 1. In this case also, SEDR performs worse than SQR at low values of ρ . However, the differences in performance are not as high as configuration 1, since the nodes in the system get more or less equally utilized, whereas in the configuration 1, nodes 1 and 2 are not at all utilized. The

observations mentioned above can be summarized to the following:

1. Policy B2 performs worse than policy B1.
2. SQR performs even better than SEDR at very low values of ρ for the configurations having service centers with widely varying service rates.
3. SEDR outperforms all other policies for higher values of ρ .

3.3 Accuracy of IQP Models

It is reasonable to expect that the accuracy of the IQP models will depend only on the interaction between the PFQN portion and the GSPN portion of the model. In particular, in the case of central server models, the parameters q_0 and ρ will influence the accuracy of the IQP models. Extensive numerical investigation has shown that the estimates produced by IQP modelling are remarkably accurate for the central server model. Even for very low values of q_0 and ρ , for which the error was the highest, the errors turned out to be very small and were within 3%. Table 3 shows for different populations, the errors in average response time values obtained using an exact GSPN model for the entire system and using the IQP modelling technique.

population of network	%error in Mean response time SQR Policy ($q_0 = 0.1; \rho = 0.2$)	%error in mean response time SEDR Policy ($\lambda = 5, \mu_1 = 1, \mu_2 = 2, \mu_3 = 3, \mu_4 = 4$)
4	2.700	2.717
5	1.244	0.975
6	0.431	0.316
7	0.143	0.105
8	0.146	0.034
9	0.015	0.011
10	0.0048	0.004
11	0.00155	0.0012

Table 3: Percentage error in mean response time values.

4 Summary

In this paper, we have applied an efficient analytical methodology for the evaluation of two heuristic dynamic load balancing policies namely shortest queue routing and shortest expected delay routing in distributed computing systems. To investigate the efficacy of this approach, we developed integrated analytical

models for the closed central server systems employing these heuristic policies. We compared the performance of these two policies with simple policies which can be modelled exactly using product form queueing networks at different levels of ρ , the imbalance in load and developed results on their relative performance.

For the central server models considered in the paper, extensive numerical results have shown that the hybrid technique employed yields remarkably accurate results. More general models have to be investigated, however. The technique can also be applied to open queueing network models in which the non-product form subnetwork has bounded population.

References

- [1] M.Ajmone Marsan, G.Balbo and G.Conte, A Class of Generalized Stochastic Petri Nets for the Performance Evaluation of Multiprocessor Systems, **ACM Transactions of Computer Systems**, Vol.2, No.2, May 1984, pp.93-122.
- [2] I.F.Akyildiz, Central Server Models with Multiple Job Classes, State Dependent Routing, and Rejection Blocking, **IEEE Transactions on Software Engineering**, Vol.15, No.10, October 1989, pp.1305-1312.
- [3] G.Balbo, S.C.Bruell and S.Ghanta, Combining Queueing Networks and Generalized Stochastic Petri Nets for the Solution of Complex Models of Computer Systems, **IEEE Transactions On Computers**, Vol.17, No.10, October 1988, pp.1251-1268.
- [4] S.A.Banawan and J.Zahorjan, Load Sharing in Heterogeneous Queueing Systems, **Proceedings of the IEEE INFOCOM'89**, 1989, pp.731-739.
- [5] J.P.C.Blanc, A Note on Waiting Times in Systems with Queues in Parallel, **Journal of Applied Probability**, Vol.24, 1987, pp.540-546.
- [6] F.Bonomi and A.Kumar, Adaptive Optimal Load Balancing in a Heterogeneous Multiserver System with a Central Job Scheduler, **IEEE Transactions on Computers**, Vol.39, No.10, October 1990, pp.1232-1250.
- [7] J.P.Buzen, Computational Algorithms for Closed Queueing Networks with Exponential Servers, **Communications of the ACM**, Vol.16, No.9, September 1973, pp.527-531.
- [8] L.Flatto and H.P.Mckean, Two Queues in Parallel, **Communications on Pure and Applied Mathematics**, Vol.30, 1977, pp.255-263.

- [9] B.W.Conolly, An Autostrada Queueing Problem, **Journal of Applied Probability**, Vol.21, 1984, pp.394-403.
- [10] D.L.Eager, D.Edward, E.D.Lazowska, and John Zahorjan, Adaptive Load Sharing in Homogeneous Distributed Systems, **IEEE Transactions on Software Engineering**, Vol.12, No.5, May 1986, pp.662-675.
- [11] G.Foschini and J.Salz, A Basic Dynamic Routing Problem and Diffusion, **IEEE Transactions on Communications**, Vol.26, No.3, March 1978.
- [12] J.A.Gubner, B.Gopinath, and S.R.S.Varadhan, Bounding Functions of a Markov Process and the Shortest Queue Problem, **Technical Report**, Department of Computer Science, University of Maryland, U.S.A., 1988.
- [13] S.Halfin, The Shortest Queue Problem, **Journal of Applied Probability**, Vol.22, 1985, pp.865-878.
- [14] J.F.C.Kingman, Two Similar Queues in Parallel, **Biometrika**, Vol.48, 1961, pp.306-310.
- [15] G.Knessl, B.J.Mattowsky, Z.Schuss and C.Tier, Two Parallel M/G/1 Queues where Arrivals Join the System with the Smaller Buffer Content, **IEEE Transactions on Communications**, Vol.35, No.11, November 1987, pp.1153-1158.
- [16] A.E.Krzesinski, Multiclass Queueing Networks with State Dependent Routing, **Performance Evaluation**, Vol.7, No.2, June 1987, pp.125-145.
- [17] R.D.Nelson and T.K.Philips, An Approximation to the Response Time for Shortest Queue Routing, **Performance Evaluation Review**, Vol.17, No.1, May 1989, pp.181-189.
- [18] M.Reiser and S.S.Lavenberg, Mean Value Analysis of Closed Multichain Queueing Networks, **Journal of ACM**, Vol.27, No.2, April 1980, pp.313-322.
- [19] .R.Meenakshi Sundaram, **Integrated Analytical Models for Parallel and Distributed Systems**, M.S.Thesis, Department of Computer Sciene and Automation, Indian Institute of Science, Bangalore, October 1990.
- [20] A.N.Tantawi and D.Towsley, Optimal Static Load Balancing in Distributed Computer Systems, **Journal of the ACM** , Vol.32, No.2, April 1985, pp.445-465.

- [21] D.Towsley, Queueing Models with State Dependent Routing, **Journal of the ACM**, Vol.27, No.2, April 1980, pp.323-337.
- [22] Yuan Chow and Walter H.Kohler, Models for Dynamic Load Balancing in a Heterogeneous Multiprocessor System, **IEEE Transactions on Computers**, Vol.28, No.5, May 1979, pp.354-361.
- [23] T.Yung, Wang and R.J.T.Morris, Load Sharing in Distributed Systems, **IEEE Transactions on Computers**, Vol.34, No.3, March 1985, pp.204-217.