A Fair Workload Allocation Policy for Heterogeneous Systems

Leonidas Georgiadis
Aristotle University
Faculty of Engineering
Dept. of Electr. Engin.
Thessaloniki
GREECE
leonid@eng.auth.gr

Christos Nikolaou
Computer Science Dept
University of Crete
Heraklion, Crete
GREECE
nikolau@csd.uch.gr

Alexander Thomasian
Computer Science Dept.
New Jersey
Inst. of Technology
Newark NJ 07102
USA
athomas@cs.njit.edu

Abstract

We consider a new workload allocation policy addressing fairness for user level performance measures. More specifically the criterion used for optimal workload allocation is the one which minimizes the maximum expected response time at computer systems to which jobs are routed. The policy to attain this criterion is therefore referred to as the min-max policy. It is shown that this optimization criterion is tantamount to routing to the fastest M processors, where M depends on system statistics, and equalizing the expected response times on these processors. The algorithm to compute job routing probabilities is applicable to increasing continuous functions of system response time versus the job arrival rate. We next investigate some properties of the minimax policy and show that it results in minimizing the coefficient of variation of response time when the job processing times are exponentially distributed. We compare the min-max policy with the one that minimizes the mean overall response time. It is shown that the new policy attains fairness by equalizing the mean response times at different systems, at a tolerable increase in overall response time. Finally, we report on a sensitivity analysis with respect to changes in job arrival rate and errors in estimating this rate.

1 Introduction

Workload allocation (or routing) is an important factor affecting the performance of distributed systems.[1] This is because the performance of such systems is not determined solely by the processing capacity of it computer systems, but rather by how well their use is coordinated. Proper workload allocation is a key factor in achieving improved performance in distributed systems. There has been a lot of activity in this area and we review relevant research in Section 2, in order to put this work in the proper perspective.

We are interested in the issue of routing jobs to the nodes of a multicomputer system, where different computers systems exhibit different job processing times. This may be due

[1] We use workload allocation instead of load sharing or load balancing on purpose. A load balancing policy strives to equalize the load at the nodes of the distributed system, while a load sharing policy strives to assure that no node is idle while there are waiting jobs. In our context workload allocation pertains to the distribution of workload to the nodes of the systems so as to satisfy certain performance objectives.

to the fact that the computers are heterogeneous or because each computer has an inherent load aside from the jobs that are being routed. The inherent load cannot be controlled (rerouted) and contends for resources with routed jobs. Our optimization criterion is to minimize the maximum average job response time at the computers to which jobs are routed (note that, depending on load, some of the slower computers may be excluded). We therefore refer to this policy as Min-Max Policy (MMP). This is shown in Section 4 to be tantamount to equalizing the average response times at a subset of the computers selected for workload allocation.

Jobs arrive at a single router which routes jobs to a number of computer systems according to a *probabilistic* routing policy (see Figure 1). The set of probabilities used for workload allocation are computed using the algorithm given in Section 4. The algorithm is applicable when the response time at each compuer system, as a function of the arrival rate, is continuous and strictly increasing. No convexity assumptions are necessary for this function.

The obvious question to answer is how does MMP compare to Minimum (average) Response time Policy (MRP), which has been used in several studies. This policy also uses probabilistic routing and optimizes the average overall job response time, i.e., the average response time seen by a job that arrives to the router - see Section 3 for the exact definition. As will be shown in Section 5.4 this policy may be unfair since the average response time at the slower computers can be much higher than the average response time at the faster ones. The MMP policy solves this problem at a tolerable increase in average overall response time.

The paper is organized as follows. In Section 2 we survey related work. In Section 3 we describe the system model under consideration. In Section 4 we derive the MMP policy. In Section 5 we study the properties of MMP and compare it with MRP. Section 6 presents a sensitivity analysis of the system with respect to variations in job arrival rate. Conclusions and proposals for future work appears in Section 7. The Appendix contains proofs of propositions used in the derivation of the MMP policy.

2 A Brief Survey of Previous Work

There is a very large body of work in the area of scheduling jobs or tasks in a system consisting of multiple computers. We do not attempt a complete review of the relevant literature here. Instead, we discuss research efforts that put the work in this paper into perspective. More related work can be found in the references cited below.

Multicomputer scheduling can be distinguished into two broad categories [15].

- *Single program task scheduling and mapping.* In this case, a single program is partitioned into a number of interdependent tasks. The objective is to allocate tasks to computers at appropriate times so that certain performance objectives - most often program completion time - are optimized. Scheduling algorithms in this class can be further partitioned in the following categories. Static, where task execution times and dependencies are known a priori and all scheduling decisions can be done offline, and dynamic where a priori information is not available and scheduling has to be done on the fly according to current system state. A thorough review of static scheduling algorithms can be found in [15]. A unified framework for dynamic load balancing has

been presented in [22]. Promising genetic algorithm-based techniques for scheduling and mapping have been also been proposed - see [6] and the references therein. An interesting and largely unexplored area of research, imposing new constraints and modeling assumptions is multiagent computing [5].

- *Job scheduling.* In this case, independent jobs arrive at the scheduler (router), whose task it to allocate (route) the jobs to the computers. The actions of the scheduler and the related algorithms depend on the available information regarding the state of the system at the time of job arrival, as well as the job resource requirements - e.g., job execution time. In addition, scheduling actions depend on the performance objective (job response time, throughput etc.) whose optimization is sought.

  When the state of the system upon job arrival is known and the computer systems are identical, and under certain assumptions on job processing times, early work has shown that the policy that routes a job to the computer with the shortest queue, satisfies several optimization objectives [21], [19]. However, as show in [20], these results are sensitive to the distribution of job processing times: as the variability of job processing time increases, optimality is lost. A comparison of several scheduling policies for a system where processors are identical and jobs have highly variable processing times has been presented in [9]. For heterogenous systems, several heuristics are proposed and compared in [14] and [2].

  When the state of the system upon job arrival in not known, then scheduling must be based only on information obtained a priori. Such information may be statistics on job arrival rates, job processing times on various processors etc. From our perspective, we can subdivide the policies in this case in two classes.

  1. *Probabilistic routing.* A fraction of jobs are routed to each computer system according to Bernoulli trials. The probability of being routed to each system is pre-computed so as to optimize a certain performance measure, as for example.

     a. *Naive Policy (NP): Route jobs in proportion to computer speeds.* This policy attempts to make all computers equally utilized. However, it has been shown that it has undesirable performance in several cases [12].

     b. *Minimum Response Policy (MRP): Minimize the average overall response time.* This criterion has been considered in several studies, [4], [17], [18], [10], [3], [16].

     c. *Min-Max Policy (MMP): Minimize the maximum response time on all computers selected for routing.* This is the policy proposed and analyzed in this study. The overall job response time will be higher in this case as compared to the MRP policy. On the other hand the MMP policy is fair in that, as will be seen, the difference in average response times at the fastest and slowest systems is eliminated. Furthermore, this is achieved at the expense of a small increase in the average overall reponse time.

  2. *Deterministic routing.* In this case jobs are assigned to computers according to a predetermined pattern, rather than probabilisticaly. An example of such a policy is the round-robin routing scheme. This scheme was shown to be optimal [19] for the case of two identical computers provided that their initial state is identical

(e.g., both are idle). However, such a simple policy cannot work well when the system consists of heterogenous computers. In the latter case, one must employ some type of weighted round robin scheme. Hence the problem of determining the appropriate weights arises. One possible approach to this problem is to employ the methodology of probabilistic routing and use the resulting routing probabilities as weights for the deterministic policy. Such an approach was used in [16] A deterministic routing policy that apportions jobs at each computer system according to prespecified weights can also be found in [8].

Before closing this section, we mention that a similar terminology to ours, namely Max-Min Policy has been used before in [14] in a very different context. The Max-Min Policy in [14], implemented in SmartNet [7], assumes knowledge of both system state and job processing times. The policy finds for each job available for routing, the computer on which it will have the minimum response time and then among these jobs selects the one whose minimum response time is maximal.

3 The Model of the Heterogeneous System

There are N computer systems each having a single or multiple processors. Jobs arrive at the job router (see Figure 1) according to a Poisson process with rate λ jobs/second. The router sends a job for execution to system S_i with probability $p_i(\lambda)$. Hence, the arrival process at S_i is Poisson with rate $\lambda_i = \lambda p_i(\lambda)$. We refer to $p_i(\lambda)$, $i = 1, 2..., N$ as the *allocation* or *routing* probabilities. The response time of a job at system S_i is defined as the length of time from the instant the job arrives to S_i, to the instant the job completes and exits the system. The Response Time Function (RTF) $R_i(x)$, specifies the average response time of a job at S_i for a job arrival rate x to that system. Provided that $p_i(\lambda)$ and $R_i(x)$ are given, we can compute the average response time of a job that arrives to the router as

$$R(\lambda) = \sum_{i=1}^{N} R_i(\lambda p_i(\lambda)) p_i(\lambda).$$

We refer to $R(\lambda)$ as the "average overall response time". Note that we assume that routing is instantaneous and does not add to response time. This assumption is not essential but simplifies the discussion.

The average processing time of a job at S_i, i.e., the average time needed to execute the job as system S_i at the absence of other jobs, is denoted by β_i. The maximum job arrival rate that S_i can sustain without the system becoming saturated is θ_i (see the discussion below). We make the following assumptions about the RTF's:[2]

Assumptions about $R_i(x)$.

1. $R_i(x)$ is a nonnegative, strictly increasing function of x.
2. $R_i(x)$ is a continuous function of x for $x \in (0, \theta_i)$.
3. $\lim_{x \searrow 0} R_i(x) = \beta_i > 0$.

[2] In what follows $x \searrow \ell$ means "as x approaches ℓ from above". Similarly, $x \nearrow \ell$ means "as x approaches ℓ from below".

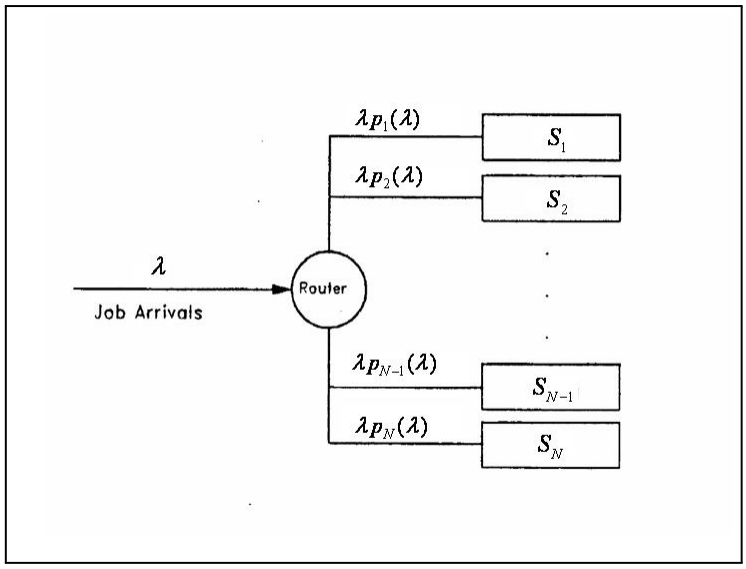

Figure 1: Multicomputer system with a central job router.

4. $\lim_{x \nearrow \theta_i} R_i(x) = \infty$.

5. $R_i(0) = 0$.

The RTF can be computed analytically or can be obtained experimentally through simulation or benchmarking. Note that no assumption about differentiability or convexity of the response time functions have been made. Assumption 3 is natural in the sense that the average processing time of a job is in fact the job response time when the arrival rate to the system is approaching zero. Assumption 4 states that the system becomes saturated (i.e., its average response time increases to infinity) as the arrival rate approaches θ_i. Assumption 5 is a convention we make to simplify the discussion.

In general we can interpret $R_i(x)$ as the cost experienced by jobs that are processed at S_i, when the arrival rate to this system is x. Moreover, Assumption 4 can be replaced by: $\lim_{x \nearrow \theta_1} R_i(x) = R_{max} \leq \infty$. For example, in place of the RTFs $R_i(x)$ we can use as cost functions the probabilities $\Pr[r_i(x) \geq \alpha]$, $i = 1, 2, ..., N$, where $r_i(x)$ is the random variable representing the steady state response time of a job processed by S_i when the arrival rate is x and α is a finite constant. In this case $\lim_{x \nearrow \theta_i} \Pr[r_i(x) \geq \alpha] = 1$, since $r_i(x)$ increases to infinity as the arrival rate approaches θ_i. Also, $\lim_{x \searrow 0} \Pr[r_i(x) \geq \alpha]$ is the probability that the processing time of a job at system S_i is at least α.

Computern system S_i will be called "faster" than S_j if $\beta_i \leq \beta_j$. Note that this does not necessarily mean that the average response times at S_i are less then those incurred at S_j for all job arrival rates. The functions $R_i(x)$ and $R_j(x)$ may intersect at some point., i.e., it may happen that $\beta_i \leq \beta_j$ and $\theta_i \leq \theta_j$. An example is the following: S_i and S_j correspond to an M/M/1 and M/M/2 queueing system respectively, such that $\beta_i < \beta_j < \beta_i \times 2$. It follows from the fact that the utilization of each system cannot exceed one, that $\theta_i = (1/\beta_i) < (2/\beta_j) = \theta_j$ (see Figure 2). Although S_i is faster than S_j (initially), there is a crossover point after which S_j becomes faster. Nevertheless, by our definition, S_i is considered to be the faster of the two systems.

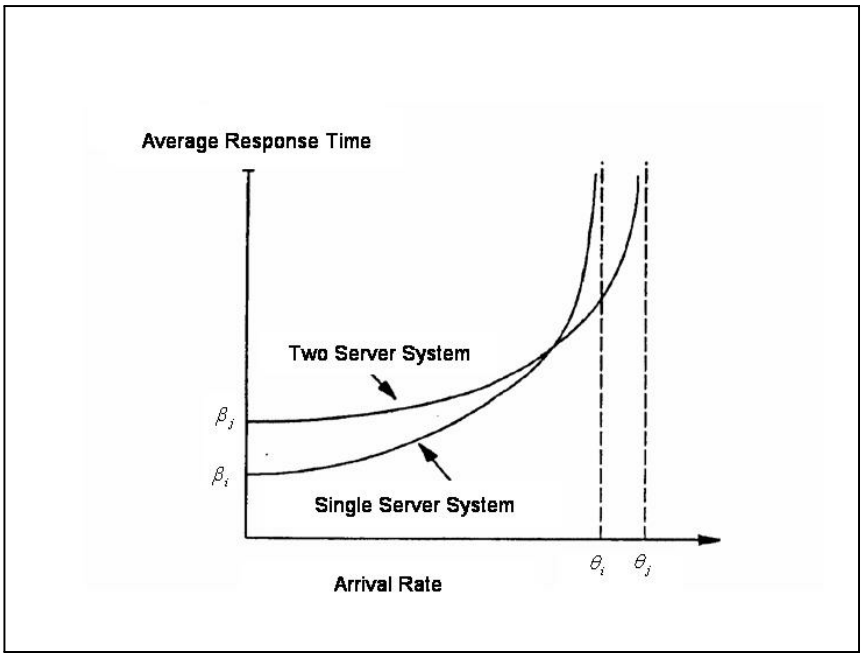

Figure 2: Example of response time functions.

The notation used in this paper is summarized in Table 1 for reader convenience. Some of the quantities in the table are introduced in the following sections.

4 Problem Formulation and Solution

The load allocated to the system that exhibits the maximum average response time is the one that is maximally penalized by the allocation policy. Since our objective is to treat every portion of the load fairly, *keeping the job response times as small as possible*, we attempt to find the policy that minimizes the average response time of the maximally penalized portion of the load among those that are routed to any of the N systems. According to our

Notation	**Definition**
λ	Job arrival rate.
β_i	Average job processing time at system S_i.
$R_i(x)$	Average job response time at S_i when the arrival rate is x.
$R(x)$	Average overall response time.
$E(x)$	Average overall response time of MMP policy
θ_i	The arrival rate at which system S_i becomes saturated.
$p_i(\lambda)$	Routing probability to S_i when job arrival rate is λ.
λ_i	Job arrival rate at system S_i ($\lambda_i = \lambda p_i(\lambda)$).
A_i	Activation rate for system S_i.

Table 1: Summary of Notation

definitions, the portion $p_i(\lambda)$ of the load that is routed to system S_i, is incurring average response time $R_i(\lambda p_i(\lambda))$. Hence we would like to keep $R_i(\lambda p_i(\lambda))$, for all $i = 1, 2...N$, as small as possible. These considerations lead to the following formulation of the criterion of optimality:

Criterion of Optimality

For a given $\lambda > 0$, find the probabilities $p_i(\lambda)$, $\imath = 1, 2, \cdots, N$, so that the maximum average response time incurred on any system is minimized:

$$\min \max_i \{R_i(\lambda p_i(\lambda))\} < \infty$$

where $\sum_{i=1}^{N} p_i(\lambda) = 1$, $p_i(\lambda) \geq 0$, $1 \leq \ i \leq N$.

In what follows we assume, without loss of generality, that the systems are indexed in nondecreasing order of their average processing times, i.e., $\beta_1 \leq \beta_2 \leq \cdots \leq \beta_N$. Given that jobs are routed only to the K fastest systems, $K < N$, it follows that $p_i(\lambda) = 0$, $K + 1 \leq i \leq N$.

In Proposition 1 we show that a policy that distributes the load among the K fastest systems, so that the response times at all these K systems are equalized, is the unique policy that satisfies the criterion of optimality. Next, in Proposition 2, we show that such a policy exists whenever the job arrival rate is less than the maximum system throughput ($\lambda < \sum_{i=1}^{N} \theta_i$). These properties will allow us to design a simple algorithm for determining the optimal routing probabilities. It will be convenient for the description of the proposition to define an additional quantity, $\beta_{N+1} = \infty$.

Proposition 1 *Let there be an integer K, $1 \leq K \leq N$ and a vector of routing probabilities*

$$\mathbf{p} = (p_1, \cdots, p_K, 0, \cdots, 0),$$

such that $R_i(\lambda p_i) < \infty$, $i = 1, ..., N$ and

$$R_1(\lambda p_1) = R_2(\lambda p_2) = \cdots = R_K(\lambda p_K) = E(\lambda) \leq \beta_{K+1}.$$

Then $\mathbf{p}$ is the unique vector satisfying the criterion of optimality.

Proof. Since $\mathrm{R}_l(\lambda p_l) = E(\lambda) \geq 0$, $1 \leq l \leq K$ and $R_i(0) = 0$, we have,

$$E(\lambda) = \max \{R_1(\lambda p_1), \cdots, \ R_K(\lambda p_K), R_{K+1}(0), ..., R_N(0)\}. \tag{1}$$

Hence $E(\lambda)$ is value of the optimization objective function when the routing probabilities are $p_i(\lambda)$, $i = 1, 2..., N$. We will show that under any other different routing probability vector, the value of the objective function exceeds $E(\lambda)$.

Let $\widehat{\mathbf{p}}$=($\widehat{\mathrm{p}}_1, \cdots, \widehat{\mathrm{p}}_\mathrm{N}$) be a routing probability vector different than $\mathbf{p}$, and let

$$\widehat{E}(\lambda) = \max \{R_1(\lambda \widehat{p}_1), \cdots, \ R_N(\lambda \widehat{p}_N)\}$$

It is sufficient to show that $\widehat{\mathrm{E}}(\lambda) > E(\lambda)$. If $\widehat{\mathrm{p}}_i > 0$ for $i \geq K+1$, then because of Assumptions 1 and 3, and the fact that $\beta_1 \leq \beta_2 \leq ... \leq \beta_{N+1}$, we have,

$$\mathrm{R}(\lambda \widehat{p}_i) > \beta_i \geq \beta_{K+1} \geq R(\lambda p_j) = E(\lambda),\ 1 \leq j \leq K.$$

From these inequalities we conclude,

$$\widehat{\mathrm{E}}(\lambda) \geq \mathrm{R}(\lambda \widehat{p}_i) > E(\lambda).$$

Assume now that $\widehat{\mathrm{p}}_{K+1} = \cdots = \widehat{p}_N = 0$. Since $\mathbf{p}$ and $\widehat{\mathbf{p}}$ are different and

$$\sum_{i=1}^{K} p_i = \sum_{i=1}^{K} \widehat{p}_i = 1,$$

it should be true that $\widehat{\mathrm{p}}_l > p_l$ for some l such that $1 \leq l \leq K$. Taking into account that $R_l(x)$ is strictly increasing we have,

$$\widehat{\mathrm{E}}(\lambda) \geq R_l(\lambda \widehat{p}_l) > R_l(\lambda p_l) = E(\lambda).$$

■

Proposition 2 *For any N, a unique routing probability vector $\mathbf{p}(\lambda)$, satisfying the conditions of Proposition 1, exists, if and only if $0 < \lambda < \sum_{i=1}^{N} \theta_i$. Furthermore, the function $E(\lambda)$ is strictly increasing, continuous in $\left(0, \sum_{i=1}^{N} \theta_i\right)$ and the following conditions hold:*

$$\begin{aligned} &a. \quad \lim_{\lambda \searrow 0} E(\lambda) = \beta_1, \\ &b. \quad \lim_{\lambda \nearrow \sum_{i=1}^{N} \theta_i} E(\ \lambda) = \infty. \end{aligned}$$

The proof of Proposition 2 is lengthy and is given in the Appendix.

Note that since the average response times at the activated systems are equalized, $E(\lambda)$ is the average overall response time under the MMP policy, i.e., $R(\lambda) = E(\lambda)$.

Based on Propositions 1 and 2 we can conclude the following. For a given λ, there is a number K_λ such that the first K_λ fastest processors are activated, i.e. the routing probabilities to these processors are nonzero. As the job arrival rate λ increases from 0 to $\sum_{i=1}^{N} \theta_i$, the number of activated systems increases from 1 to N. For $\lambda > \sum_{i=1}^{N} \theta_i$ the system is saturated. System S_K is activated when λ exceeds a threshold arrival rate A_K, which will be called the *activation rate for S_K*. The activation rates have the following properties: [3]

1. $A_1 = 0$
2. $A_1 \leq A_2 \leq \cdots \leq A_{N-1} \leq A_N < \sum_{i=1}^{N} \theta_i$
3. $R(A_k) = \beta_k$.
4. $A_k = A_l$ when $\beta_k = \beta_l$
5. When $\lambda \leq A_k$, the traffic is distributed among the systems that have smaller average processing times than S_k, so that the average response times induced on all active systems are equalized and are at most β_k.

[3] Note that the systems are ordered according to nondecreasing processing times.

Properties 1 and 4 imply that the fastest systems, i.e., those with $\beta_i = \beta_1$, are always activated (as long as the load is non-zero). Property 2 means that the ordering of activation rates is the same as the order of the systems. Property 3 stands for the fact that at the activation rate of S_k the average overall response time is equal to the job processing time at system S_k.

The activation rates and the routing probabilities at those rates can be easily computed as follows. Let $R_i^{-1}(r)$ denote the inverse of $R_i(\lambda)$, i.e., $R_i^{-1}(r)$ is the arrival rate that induces average response time equal to r on S_i. Let M_K be the number of systems that are strictly faster than S_K. Since, by Property 3, the average response time on $S_i, 1 \leq i \leq M_K$ is equal to β_K when $\lambda = A_K$ and $\lambda_i = A_K p_i(A_K)$, we conclude that

$$A_K p_i(A_K) = R_i^{-1}(\beta_K), 1 \leq i \leq M_K. \tag{2}$$

Since $\sum_{i=1}^{M_K} p_i(\ A_K) = 1$, by summing the equation in (2), it follows that

$$A_k = \sum_{i=1}^{M_K} R_i^{-1}(\beta_K). \tag{3}$$

From (2) and (3) we conclude,

$$p_i(A_K) = \frac{R_i^{-1}(\beta_K)}{A_K} = \frac{R_i^{-1}(\beta_K)}{\sum_{i=1}^{M_K} R_i^{-1}(\beta_K)},\ 1 \leq i \leq M_K. \tag{4}$$

From (3) we compute the activation rate for system S_K, and from (4) we compute the corresponding routing probability vector.

Next we describe an algorithm by which the allocation vector and $R(\lambda)$ can be computed for an arbitrary λ. Assume first that $A_{K-1} < \lambda < A_K$, $2 \leq K \leq N$. Then the M_K fastest systems will be activated. For a specified overall average response time r, we can obtain the arrival rate λ_r, and the routing probability vector $\mathbf{p}(\lambda_r)$, that induces the specified response time. Indeed, following the reasoning by which equations (3) and (4) were obtained, we have that:

$$\lambda_r = \sum_{i=1}^{M_K} R_i^{-1}(r), \tag{5}$$

and

$$p_i(\lambda_r) = \frac{R_i^{-1}(r)}{\lambda_r} = \frac{R_i^{-1}(r)}{\sum_{i=1}^{M_K} R_i^{-1}(r)},\ i = 1, 2, ...M_K. \tag{6}$$

If $\lambda_r < \lambda$, then since $R(\lambda)$ is increasing (see Proposition 2) we conclude that $r < R(\lambda)$, i.e., r is a lower bound on $R(\lambda)$. Similarly, if $\lambda_r > \lambda$, then $r > R(\lambda)$, i.e. r is an upper bound on $R(\lambda)$. Therefore, if initial upper and lower bounds R_u, R_l on $R(\lambda)$ are known, we can obtain

$R(\lambda)$ by a simple binary search on the interval $[R_l, R_u]$. That is, we try $r = (R_l + R_u)/2$. If $\lambda_r < \lambda$ then we set $R_l \leftarrow r$ and repeat the process. If $\lambda_r > \lambda$ then we set $R_u \leftarrow r$ and repeat the process. The initial upper and lower bounds are: $R_l = \beta_{K-1}, R_u = \beta_K$. The process ends whenever

$$\max_{1 \le i \le M_K} \{R_i(\lambda p_i(\lambda_r))\} - \min_{1 \le i \le M_K} \{R_i(\lambda p_i(\lambda_r)\} < \varepsilon,$$

where $\varepsilon > 0$ is a small constant, and the routing probability vector is $p_i(\lambda_r), i = 1, 2, ..., M_K$.

Next assume that $A_N < \lambda < \sum_{i=1}^{N} \theta_i$. Then, all systems are activated. In this case, we can simply test successively increasing values of r_n. (e.g. $r_{n+1} = 2r_n, r_1 = \beta_N$), until $\lambda_{r_n} \ge \lambda$, in which case, $R_u = r_n$, and $R_l = r_{n-1}$. Then, we can perform the standard binary search. The steps of the algorithm are described below.

Algorithm. Compute the Allocation Vector $\underline{p}(\lambda)$

Input: The response time functions $R_i(\lambda)$, $1 \le i \le N$ and the job arrival rate λ.
Output: The routing probability vector $\mathbf{p}(\lambda)$.

1. ***Compute activation rates***
   $M_K \leftarrow$ number of S_i' s with $\beta_i < \beta_K, 1 \le K \le N$

   $A_K = \sum_{i=1}^{M_K} R_i^{-1}(\beta_K), 1 \le K \le N.$

2. ***Determine the number of active systems, at rate*** λ.
   If $\lambda \ge \sum_{i=1}^{N} \theta_i$ then stop /*solution impossible*/
   If $A_{K-1} < \lambda \le A_K$, define $J \leftarrow M_K$; else $J \leftarrow N$;

3. ***Initial Upper and Lower Bounds*** (R_u and R_l)

   If $\lambda = A_K$ then $R_l = R_u = \beta_K$. Else,

   If $J < N$ then $R_l \leftarrow \beta_{K-1}; R_u \leftarrow \beta_K$. Else do
   $R_l \leftarrow \beta_N; R_u \leftarrow 2 \times R_l$
   /*Test increasing values for R_u until $\lambda \le \lambda_{R_u}$*/
   Until $\lambda \le \sum_{i=1}^{N} R_i^{-1}(R_u)$ do
   $R_l \leftarrow R_u; R_u \leftarrow 2 \times R_l$
   end
   end

4. ***Initialize iteration***
   $r \leftarrow (R_l + R_u)/2$
   $p_i = R_i^{-1}(r) / \sum_{i=1}^{J} R_i^{-1}(r)$, $1 \le i \le J$

5. ***Iterate until convergence criterion is satisfied***
   Until $(\max_{1 \le i \le M_K}(R_i(\lambda p_i)) - \min_{1 \le i \le M_K}(R_i(\lambda p_i))) < \varepsilon$ do

   Determine new bounds
   if $\lambda < \sum_{i=1}^{M_K}(R_i^{-1}(r))$ then $R_u \leftarrow r$; else $R_l \leftarrow r$

Compute average of new bounds
$r \leftarrow (R_u + R_l)/2$

Compute the allocation vector that induces response time r
$p_i = R_i^{-1}(r)/\sum_{i=1}^{J} R_i^{-1}(r)$, $1 \le i \le J$

end

6. *Return* p_i, $1 \le i \le J$.

End of Algorithm

5 Properties of MMP and Comparison with MRP

In this section we first examine some interesting properties of MMP and we then proceed to compare this policy to MRP.

5.1 Coefficient of Variation of Mean Response Time for M/M/1 Systems

When the systems are modeled as M/M/1 queues, MMP has the additional property of minimizing the coefficient of variation of the average overall response time under all probabilistic routing policies. To see this note that under a probabilistic policy, system S_i, $i = 1, 2, ..N$, behaves like an M/M/1 queue. It is well known [11] that in this case the variance σ_i^2 of the response time of a job routed to S_i is equal to R_i^2, where R_i is the average response time of a job at S_i: $R_i = R_i(\lambda)$. Therefore, under a probabilistic policy using the routing probabilities p_i, $i = 1, 2, ..N$, the variance of the overall average response time is given by (we use the fact that for a random variable X, $\sigma_X^2 = E[X^2] - (E[X])^2$):

$$\sigma_{total}^2 = \sum_{i=1}^{N} (\sigma_i^2 + R_i^2) p_i - \left(\sum_{i=1}^{N} R_i p_i \right)^2 = 2 \sum_{i=1}^{N} R_i^2 p_i - \left(\sum_{i=1}^{N} R_i p_i \right)^2 .$$

The coefficient of variation c_v, of the average overall response time is equal to:

$$c_v = \sqrt{\frac{\sigma_{total}^2}{\left(\sum_{i=1}^{N} R_i p_i \right)^2}} = \sqrt{\frac{2 \sum_{i=1}^{N} R_i^2 p_i}{\left(\sum_{i=1}^{N} R_i p_i \right)^2} - 1}$$

From Jensen's inequality [1] we have,

$$\sum_{i=1}^{N} R_i^2 p_i \ge \left(\sum_{i=1}^{N} R_i p_i \right)^2 ,$$

with equality holding if and only if $R_i = R_j$, whenever $p_i \neq 0$ and $p_j \neq 0$. Therefore, the minimum value of c_v is 1. Since MMP equalize R_is on the active systems, it achieves this minimum.

5.2 The Effect of Job Processing Time Variation on Activation Rates for M/G/1 Systems

Let jobs have an inherent processing requirement given by a random variable B. Assume also that system S_i has processing capacity C_i, $1 \leq i \leq N$. For example, B may correspond to the number of instructions executed per job and C_i to the MIPS rating for the processor of S_i. It follows that the average processing time of a job at S_i is $\beta_i = E[B]/C_i$.

In this section we consider the effect of variability of job processing times on workload allocation. Let B_1 and B_2 denote the processing requirements of two workloads (set of jobs) to be processed by the same system configuration consisting of N systems. We assume that $E[B_1] = E[B_2]$ so that $\beta_{1i} = \beta_{2i} = \beta_i$, $1 \leq i \leq N$. On the other hand the coefficient of variation of processing requirements is such that $c_{2v} \geq c_{1v}$, i.e., the processing requirements of the second workload have a higher variability than the first workload.

The RTF at S_i for the first and second workload is given by the average response time equation for M/G/1 queues [11],

$$R_{1i}(\lambda) = \beta_i + \frac{\lambda\beta_i^2(1+c_{1v}^2)}{2(1-\lambda\beta_i)}, 1 \leq i \leq N, \tag{7}$$

$$R_{2i}(\lambda) = \beta_i + \frac{\lambda\beta_i^2(1+c_{2v}^2)}{2(1-\lambda\beta_i)}, 1 \leq i \leq N. \tag{8}$$

The number of systems, M_K, that are active before S_K is first activated, depends only on β_i, $1 \leq i \leq N$ and hence is the same in both cases. The activation rates are computed as follows:

$$A_{1K} = \sum_{i=1}^{M_K} R_{1i}^{-1}(\beta_i) \tag{9}$$

$$A_{2K} = \sum_{i=1}^{M_K} R_{2i}^{-1}(\beta_i). \tag{10}$$

Since $c_{2v} \geq c_{1v}$, it follows from equations (7) and (8) that $R_{2i}(\lambda) \geq R_{1i}(\lambda)$. Therefore, $R_{1i}^{-1}(\beta_i) \geq R_{2i}^{-1}(\beta_i)$. From the last inequality and equations (9) and (10) we conclude that $A_{1K} \geq A_{2K}$. In other words, as the coefficient of variation increases, MMP activates the slower system at lower job arrival rates.

5.3 Activation Rates for MMP versus MRP

The problem of minimizing the average overall response time is formulated as follows:

$$\min\left\{\sum_{i=1}^{N} p_i R_i(\lambda p_i)\right\} \text{ where } \sum_{i=1}^{N} p_i = 1,\ p_i \geq 0.$$

It can be shown using the Lagrange Multiplier method (see [4] and [17]), that when the RTFs are strictly increasing, differentiable and convex, a solution with similar properties to

those of the MMP policy is obtained. The activation rates in this case are defined as follows:

$$A_K^{MRP} = \sum_{i=1}^{M_K} f_i^{-1}(\beta_i),$$

where $f_i^{-1}(\lambda)$ is the inverse of $f_i(\lambda)$ which is defined as

$$f_i(\lambda) = (\lambda R_i(\lambda))' = R_i(\lambda) + \lambda R_i'(\lambda), 1 \leq i \leq N.$$

M_K, the number of active systems when S_K is activated, is determined exactly as in MMP. Since $R_i(\lambda)$ are assumed to be convex, $R_i'(\lambda) > 0$. Therefore, $f_i(\lambda) > R_i(\lambda)$, $1 \leq i \leq N$. Using the same argument as in the previous section, we conclude that $A_K^{MRP} < A_K^{MMP}$. Therefore, MMP is more reluctant to allocate jobs to the slower processors than MRP.

5.4 Comparison of Performance for M/G/1 Systems

MRP minimizes the average overall response time, but the portions of the load allocated to slower processors may suffer excessively. MMP remedies the situation by equalizing the average response times on all the active processors, but does not minimize the average overall response time. In this section we examine in more detail the trade-offs incurred by the two policies.

Assume that we have two systems S_1, S_2, and let p_i^{MRP}, p_i^{MMP}, $i = 1, 2$, be the portion of traffic that is allocated to S_i under MRP and MMP respectively. The average response times on the two systems and the overall average response time under MRP and MMP are denoted by $R_i^{MRP}(\lambda p_i(\lambda))$, $i = 1, 2$, $R^{MRP}(\lambda)$ and $R_i^{MMP}(\lambda p_i(\ \lambda))$, $i = 1, 2$, $R^{MMP}(\lambda)$ respectively. As before, we order the systems according to their speed, hence, system S_1 is faster than system S_2, i.e., $\beta_1 < \beta_2$.

We concentrate our attention on two measures that represent the trade-offs involved in applying the two policies. First, we consider the ratio of response times at the slower and faster system for the MRP policy, provided that both systems are activated:

$$Q_r(\lambda) = \frac{R_2^{MRP}(\lambda p_2(\lambda))}{R_1^{MRP}(\lambda p_1(\lambda))}.$$

Next we consider the ratio of average overall response times for MMP and MRP,

$$Q_o(\lambda) = \frac{R^{MMP}(\lambda)}{R^{MRP}(\lambda)}.$$

Note that for the MMP policy the corresponding ratio $Q_r(\lambda)$ is always one and that since MRP minimizes the average overall response time, $Q_0(\lambda) \geq 1$.

Let us consider first the case when the processing time distribution is exponential. We can then easily derive simple closed form solutions for both MRP and MMP. We omit the straigthforward but somewhat tedious calculations. The activation rates under the two policies are given by:

$$A_2^{MRP} = \frac{1}{\beta_1} - \frac{1}{\sqrt{\beta_1 \beta_2}},$$

$$A_2^{MMP} = \frac{1}{\beta_1} - \frac{1}{\beta_2}.$$

The routing probabilities of the two policies are:

$$p_1^{MRP}(\lambda) = \begin{cases} 1 & \text{if } \lambda \leq \frac{1}{\beta_1} - \frac{1}{\sqrt{\beta_1\beta_2}} \\ \frac{\sqrt{\beta_1\beta_2} - \beta_1 + \lambda\beta_1\beta_2}{\lambda\beta_1\left(\beta_2 + \sqrt{\beta_1\beta_2}\right)} & \text{if } \frac{1}{\beta_1} - \frac{1}{\sqrt{\beta_1\beta_2}} < \lambda < \frac{1}{\beta_1} + \frac{1}{\beta_2} \end{cases}, \tag{11}$$

$$p_1^{MMP}(\lambda) = \begin{cases} 1 & \text{if } \lambda \leq \frac{1}{\beta_1} - \frac{1}{\beta_2} \\ \frac{1}{2} + \frac{1}{2\,\lambda}\left(\frac{1}{\beta_1} - \frac{1}{\beta_2}\right) & \text{if } \frac{1}{\beta_1} - \frac{1}{\beta_2} < \lambda < \frac{1}{\beta_1} + \frac{1}{\beta_2} \end{cases}. \tag{12}$$

For both policies, the average response times on each system and the average overall response time can be easily computed once the routing probabilities have been determined:

$$R_i(\lambda) = \frac{\beta_i}{1 - \beta_i\lambda p_i(\lambda)},\ i = 1, 2, \tag{13}$$

$$R(\lambda) = p_1(\lambda)R_1(\lambda) + p_2(\lambda)R_2(\lambda). \tag{14}$$

From equations (11) and (13) we find that when the second system is activated,

$$Q_r(\lambda) = \sqrt{\frac{\beta_2}{\beta_1}} = s \geq 1 \tag{15}$$

$Q_r(\lambda)$ does not depend on λ in this case and it increases as the square root of β_2/β_1. In contrast, the corresponding ratio is always equal to one under the MMP policy.

Let us now turn our attention to the performance of the two systems in terms of overall response time. Based on equations (11)-(14) we can compute $Q_o(\lambda)$. It turns out that $Q_o(\lambda)$ depends on λ and its maximum is achieved when $\lambda = (1/\beta_1 - 1/\beta_2)$, which is the point at which the second system is activated under MMP. More specifically, we have,

$$Q_o = \max_{\lambda} Q_o(\ \lambda) = \frac{2 + 2s}{3 + s} = 2 - \frac{4}{3 + s}. \tag{22}$$

We see that Q_o increases with $s = \sqrt{\beta_2/\beta_1}$, but remains bounded and never exceeds 2.

In Figures 3 and 4, we plot the various response times under the two policies for $\beta_2/\beta_1 =$ 2 and 6 respectively. It can be observed that while the average overall response times achieved by MMP and MRP are quite close to each other, there is a significant difference in response time at the two systems for MRP.

When the processing times are nonexponential, the expression for $Q_r(\lambda)$ and $Q_o(\lambda)$ are fairly complicated but numerical results can be obtained. Similar to Q_o, we define $Q_r = \max_\lambda Q_r(\lambda)$. In Tables 2 and 3 we provide the values of Q_r and Q_o, respectively, for

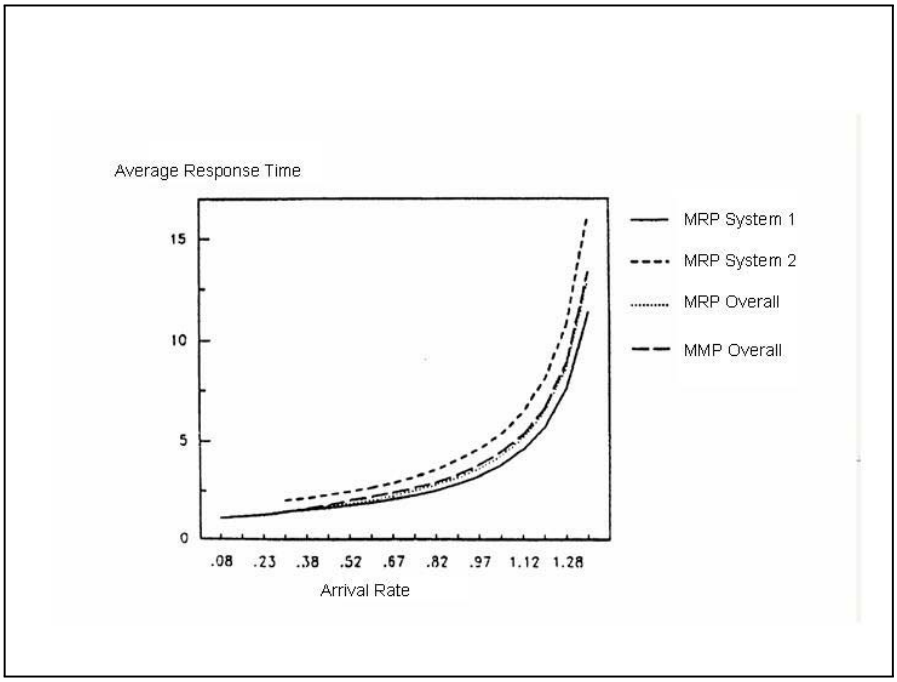

Figure 3: Mean response tiems ($\beta_1 = 1$sec, $\beta_2 = 2$sec).

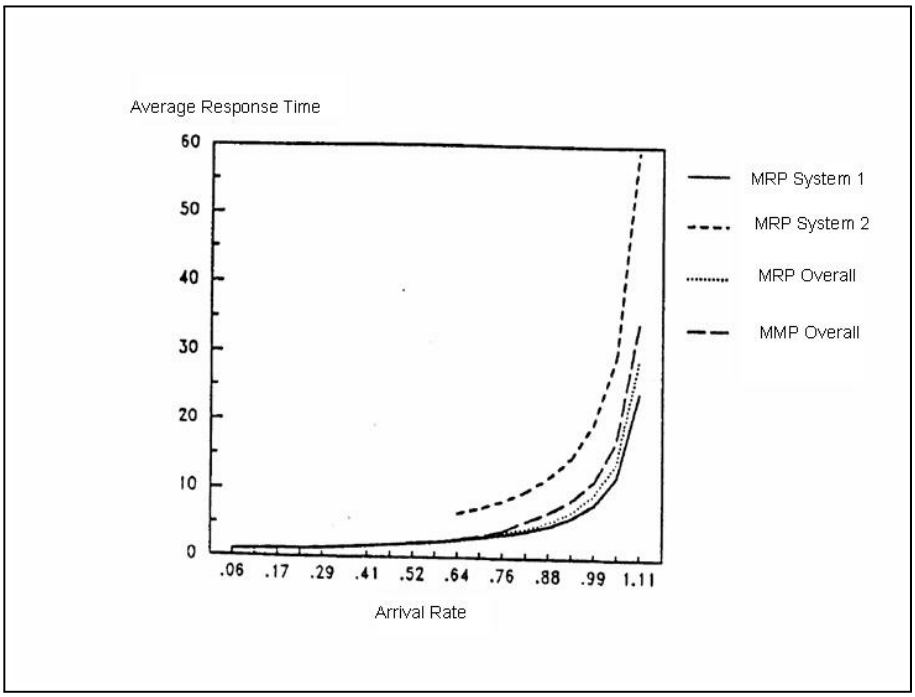

Figure 4: Mean response times ($\beta_1 = 1$sec, $\beta_2 = 6$sec).

various values of β_2/β_1 and for various values of the coefficient of variation of service time. We observe that for the values of β_2/β_1 and C_v used, the same conclusion as for the M/M/1 case holds: R_r increases approximately as the square root of β_2/β_1, while R_r increases, but remains bounded and less than 2.

β_2/β_1	$C_v = 0$	$C_v = 1$	$C_v = 4$	$C_v = 10$
2	1.50	1.41	1.40	1.40
4	2.24	2.00	1.96	1.96
6	2.83	2.44	2.38	2.38
10	3.78	3.16	3.04	3.04
16	4.92	4.00	3.78	3.75

Table 2: Q_r for two M/G/1 systems

β_2/β_1	$C_v = 0$	$C_v = 1$	$C_v = 4$	$C_v = 10$
2	1.15	1.09	1.03	1.03
4	1.37	1.20	1.10	1.10
6	1.50	1.26	1.16	1.16
10	1.69	1.35	1.23	1.23
16	1.82	1.42	1.28	1.27

Table 3: Q_o for two M/G/1 systems

6 Sensitivity of the Solution to Arrival Rate Estimates

The optimal workload allocation is based on the knowledge of job arrival rate, λ. In an operational system, however, the job arrival rate may fluctuate. Besides, its value is estimated by using some recent history and therefore, it is not known exactly. Hence it is important to know the effect of arrrival rate fluctuations and inacurate estimations on the performance of the system. In Section 5.1 we study how small changes in λ affect the job routing policy and the average overall response time $R(\lambda) = E(\lambda)$. In Section 5.2 we examine the deviation from the optimum $R(\lambda)$ due to inaccurate estimation of the job arrival rate.

6.1 Effect of Changes in Job Arrival Rate

Let us assume that the functions $R_i(\lambda)$, $1 \leq i \leq N$, are differentiable with respect to λ (this was not required in the previous section). The derivative of $R_i(\lambda)$ is denoted by $R_i'(\lambda)$. In the course of the proof of Propositions 2 in the Appendix, it was shown that $p_i(\lambda)$, $1 \leq i \leq N$, is a continuous function of λ. Using the continuity of $p_i(\lambda)$, it can be shown that $p_i'(\lambda)$, $1 \leq i \leq N$ and $R(\lambda)$, are differentiable for $\lambda \neq A_K$, $1 \leq K \leq N$. The corresponding derivatives for all values of $\lambda \neq A_K$, $1 \leq K \leq N$, can be computed as follows:
It will be convenient for the description that follows to define $A_{N+1} = \sum_{i=1}^{N} \theta_i$ and

$M_{N+1} = N$. Let $A_K < \lambda < A_{K+1}$. Then the following equalities hold (see Proposition 1):

$$R(\lambda) = R_i(\lambda p_i(\lambda)), 1 \leq i \leq M_{K+1}. \tag{16}$$

By differentiating the equations in (16) with respect to λ we have:

$$R'(\lambda) = (p_i(\lambda) + \lambda p_i'(\lambda)) \times R_i'(\lambda p_i(\lambda)),\ 1 \leq i \leq M_{K+1}, \tag{17}$$

or

$$\frac{R'(\lambda)}{R_i'(\lambda p_i(\lambda))} = p_i(\lambda) + \lambda p_i'(\lambda),\ 1 \leq i \leq M_{K+1}, \tag{18}$$

Summing equations (18), we have that

$$R'(\lambda) \sum_{j=1}^{M_{K+1}} \frac{1}{R_j'(\lambda p_j(\lambda))} = \sum_{i=1}^{M_{K+1}} p_i(\lambda) + \lambda \sum_{i=1}^{M_{K+1}} p_i'(\lambda) = 1. \tag{19}$$

The last equality follows from the fact that $\sum_{i=1}^{M_{K+1}} p_i(\lambda) = 1$ and hence $\sum_{i=1}^{M_{K+1}} p_i'(\lambda) = 0$. Therefore, the derivative of $R(\lambda)$ is given by,

$$R'(\lambda) = \frac{1}{\sum_{j=1}^{M_{K+1}} \frac{1}{R_j'(\lambda p_j(\lambda))}},\ \text{if } A_K < \lambda < A_{K+1}, K \in \{1, \ldots, N\}. \tag{20}$$

From equations (18) and (20), we finally conclude that:

$$p_i'(\lambda) = \frac{\frac{1}{R_i'(\lambda p_i(\lambda))}}{\lambda \times \sum_{j=1}^{M_{K+1}} \frac{1}{R_j'(\lambda p_j(\lambda))}} - \frac{p_i(\lambda)}{\lambda}, \text{if } A_K < \lambda < A_{K+1}, K \in \{1, \ldots, N\} \tag{21}$$

Equations (21) can be used to quickly recompute the job routing policy for small changes of the arrival rate. Specifically, if the arrival rate becomes $\lambda + \delta$, $A_K < \lambda + \delta < A_{K+1}$, then the allocation vector becomes approximately,

$$p_i(\lambda + \delta) \approx p_i(\lambda) + \delta p_i'(\lambda),\ i = 1, 2..., N.$$

6.2 Effect of Estimation Errors of Job Arrival Rate

To see the effect of estimation errors of job arrival rate on system response time, let us assume that the exact job arrival rate is λ_1, while it was estimated as λ. To simplify the discussion, we assume that $A_K < \lambda_1 < A_{K+1}$ and $A_K < \lambda < A_{K+1}$. Since the load allocation policy has been computed using the estimate λ, the maximum average response time will be

$$R(\lambda_1, \lambda) = \max_{1 \leq l \leq K} \{R_l(p_l(\lambda) \times \lambda_1)\}.$$

We have used the notation $R(\lambda_1, \lambda)$ to specify the two job arrival rates relevant to the discussion. The difference between the resulting maximum and the computed one will then be:

$$R(\lambda_1, \lambda) - R(\lambda) = \max_{1 \le j \le M_{K+1}} \{R_j(p_j(\lambda) \times \lambda_1) - R_j(p_j(\lambda) \times \lambda)\}. \tag{22}$$

In equation (22) we used the fact that $R(\lambda) = R_j(p_j(\ \lambda) \times \ \lambda), j = 1, ..., M_{K+1}$. If $\lambda_1 > \lambda$, then by dividing equation (22) by $\lambda_1 - \ \lambda$ and taking the limits when $(\lambda_1 - \ \lambda) \to 0$, it can be seen that

$$\lim_{\lambda_1 \searrow \lambda} \frac{R(\lambda_1, \lambda) - R(\lambda)}{\lambda_1 - \lambda} = \max_{1 \le j \le M_{K+1}} \left\{p_j(\lambda) \times R_j'(p_j(\lambda)\lambda)\right\} > 0. \tag{23}$$

Similarly, if $\lambda_1 < \lambda$, we conclude that

$$\lim_{\lambda_1 \nearrow \lambda} \frac{R(\lambda_1, \lambda) - R(\lambda)}{\lambda_1 - \lambda} = \min_{1 \le l \le M_{K+1}} \left\{p_l(\lambda) \times R_l'(p_l(\lambda)\lambda)\right\} > 0. \tag{24}$$

To take the limit in (24), we divide (22) by $\lambda_1 - \ \lambda$, a negative number. This explains the reason that we have a minimum on the right hand side.

Since the RTFs are increasing, their derivatives are positive, and therefore the limits in equations (23) and (24) are positive. Equations (23) and (24) indicate the deviation of the attained response time from the computed one when the job arrival rate is underestimated or overestimated, respectively. Since both limits are positive, the limit in (23) is larger than the limit in (24) in absolute value. Therefore, for small deviations from the actual job arrival rate, the deviation of the attained average overall response time from the computed one, is larger when the arrival rate is underestimated.

Equations (23) and (24) provide information about the deviation of the maximum average response time from the computed one. It is of interest, however, to know the deviation of the maximum average response time from the solution we would have obtained if the rate was correctly estimated, i.e. $R(\lambda_1)$. Using similar reasoning as in the derivations of equations (23) and (24) we find that for $A_k < \lambda < A_{K+1},\ K = 1, \ldots, N$

$$\lim_{\lambda \searrow \lambda_1} \frac{R(\lambda_1, \lambda) - R(\lambda_1)}{\lambda - \lambda_1} = \lambda_1 \times \max_{1 \le j \le M_{K+1}} \left\{p_j'(\lambda_1) \times R_j'(p_j(\lambda_1)\lambda_1)\right\} > 0. \tag{25}$$

$$\lim_{\lambda \nearrow \lambda_1} \frac{R(\lambda_1, \lambda) - R(\lambda_1)}{\lambda - \lambda_1} = \lambda_1 \times \min_{1 \le l \le M_{K+1}} \left\{p_l'(\lambda_1) \times R_l'(p_l(\lambda_1)\lambda_1)\right\} < 0. \tag{26}$$

Since $\sum_{i=1}^{M} p_l'(\ \lambda_1) = 0$, some of the derivatives $p_l'(\ \lambda_1)$ will be positive and some negative. As a result, the limit in (25) is positive and the limit in (26) is negative. Therefore, the difference $R(\lambda_1, \lambda) - R(\lambda_1)$ is positive irrespective of whether the job arrival rate is underestimated or overestimated. Of course this is to be expected, since $R(\lambda_1)$ is the average overall response time under the optimal policy. Since the limits in (25) and (26) are of opposite sign, we cannot determine in general which is greater in absolute value.

7 Conclusion

We proposed a new criterion for load balancing in distributed systems, which is based on optimizing (minimizing) a user level performance measure (average response time), while taking into account fairness. This criterion is more appealing than minimizing average overall response time, because such a policy is difficult to justify to users who encounter a long turnaround time when their job is routed to the slower computer system.

An efficient computational algorithm to obtain the routing probabilities was described. Although our examples deal with *M/G/1* type queueing systems, the proposed algorithm is applicable to more complex queueing systems, as long as the response time characteristic of the system is known either analytically or from measurements.

We also described several interesting properties of the proposed policy and compared its performance with the policy that minimizes the average overall response time.

An important extension to this work is to consider a system with multiple job types. The fairness criterion in this case should be chosen such that it takes into account the different processing requirements of each job type.

References

[1] R. B. Ash, *Real Analysis and Probability,* Academic Press, 1972.

[2] T. D. Braum, H. J. Siegel, N. Beck, L. L. Boloni, M. Maheswaran, A. I. Reuther, J. P. Robertson, M. D. Theys and B.Yao, "A Comparison of Eleven Static Heuristics for Mapping a Class of Independent Tasks onto Heterogeneous Distributed Computing Systems," *Journal of Parallel and Distributed Computing,* 61 (6),(2001), 810-837.

[3] S. C. Borst, "Optimal Probabilistic Allocation of Customer Types to Servers," *Proc. ACM SIGMETRICS '95,* Ottawa, Ontario, Canada, 116-125, 1995.

[4] J. P. Buzen and P. P. S. Chen, "Optimal Load Balancing in Memory Hierachies," *Information Processing*, North-Holland, New York, 1974, pp. 271–275.

[5] K. Chow and Y. Kwok, "On Load Balancing for Disgtributed Multiagent Computing," *IEEE Transactions on Parallel and Distributed Computing,* 13 (8), (2002), 787-801.

[6] M. K. Dhodhi, Im. Ahmad, A. Yatama and Is. Ahmad, "An Integrated Technique for Task Matching and Scheduling onto Distributed Heterogenous Computing Systems," *Journal of Parallel and Distributed Computing*, 62, (2002),1338-1361.

[7] R. F. Freung, M. Gherrity, S. Ambrosius, M. Cambell, M. Halderman, D. Hesgen, E. Keith, T. Kidd, M. Kussow, J.D. Lima, F. Mirabile, L. Moore, B. Rust, and H. J. Siegel, "Scheduling Resources in Multiuser, Heterogenous Computer Environments with SmartNet," *Proc. 7th IEEE Heterogenous Computing Workshop, (HCW '98),* 184-199,1998.

[8] B. Hajek, "Extremal Splittings of Point Process," *Mathematics of Operation Research*, 10 (4), (Nov 1985),543–556.

[9] M. Harchol-Balter, M. E. Crovella and C. D. Murta, "On Choosing a Task Assignment Policy for a Distributed Server System," *Journal of Parallel and Distributed Computing,* 59, (1999), 204-228.

[10] C. Kim and H. Kameda, "An Algorithms for Optimal Static Load Balancing in Distributed Computer Systems," *IEEE Transactions on Computers,* (41) 3, (March 1992), 381-384.

[11] L. Kleinrock, *Queueing Systems Vol 1: Theory,* Wiley Interscience, 1975.

[12] R. Leslie and S. McKenzie, "Evaluation of Loadsharing Algorithms for Heterogenous Distributed Systems." *Computer Communications,* 22(4), Mar 1999, 376-389.

[13] J. Li and H. Kameda, "Optimal Static Load Balancing in Start Network Configuration with Two-Way Traffic," *Journal of Parallel and Distributed Computing,* 23 (3), (1994), 364-375.

[14] M. Maheswaran, S. Ali, H. J. Siegel, D. Hensgen and R.F. Freud, "Dynamic Mapping of a Class of Independent Tasks onto Heterogenous Computing Systems," *Journal of Parallel and Distributed Computing,* 59 (2), (1999), 107-131.

[15] Y. Kwok and I. Ahmad, "Static Scheduling Algorithms for Allocating Directed Task Graphs to Multiprocessors," *ACM Computing Surveys,* 31 (4), (Dec. 1999), 406-471.

[16] X. Tang and S. T. Chanson, "Optimizing Static Job Scheduling in a Network of Heterogenous Computers," *Proc. of the 29th International Conference on Parallel Processing (ICPP),* 373-382, 2000.

[17] A. N. Tantawi and D. Towsley, "Optimal Static Load Balancing in Distributed Computer Systems," *Journal of the ACM*, 32 (2), (April 1985), 445–465.

[18] A. Thomasian, "A Performance Study of Dynamic Load Balancing in Distributed Sytems," in *Proc. 7th Int'l Conf. Distributed Computing Systems*, West Berlin, Germany, September 1987, pp. 204–217.

[19] P. Varaiya A. Ephremides and J. Walrand, "A Simple Dynamic Routing Problem," *IEEE Trans. Automatic Control*, vol. AC-25., 4, 690–693, August 1980.

[20] W. Whitt, "Deciding Which Queue to Join: Some Counterexamples," *Operations Research,* 34 (1), (Jan. 1986), 226-244.

[21] W. Winston, "Optimality of the Shortest Line Discipline," *Jounal of Applied Probability*, 14, (1977), 181–189.

[22] T. Znati and P. Melhem, "A Unified Framework for Dynamic Load Balancing Strategies in Distributed Processing Sytems," *Journal of Parallel and Distributed Computing,* 23 (2), (1994), 246-255.

8 Appendix

In this Appendix we prove Proposition 2. We first need to establish some useful inequalities.

Assume that we have two systems S_1 and S_2 and assume that there are two arrival rates $\lambda_1 < \lambda_2$, for which routing probabilities $p_i(\lambda_1)$ and $p_i(\lambda_2)$, $i = 1, 2$, satisfying the conditions in Proposition 1 can be found. Then the following inequalities are satisfied:

$$E(\lambda_1) < E(\lambda_2), \tag{27}$$

$$\lambda_1 p_1(\lambda_1) \leq \lambda_2 p_1(\lambda_2), \tag{28}$$

$$\lambda_1 p_2(\lambda_1) \leq \lambda_2 p_2(\lambda_2), \tag{29}$$

$$\frac{\lambda_1}{\lambda_2} p_1(\lambda_1) \leq p_1(\lambda_2) \leq \frac{\lambda_2 - \lambda_1}{\lambda_2} + \frac{\lambda_1}{\lambda_2} p_1(\lambda_1), \tag{30}$$

$$\frac{\lambda_1}{\lambda_2} p_2(\lambda_1) \leq p_2(\lambda_2) \leq \frac{\lambda_2 - \lambda_1}{\lambda_2} + \frac{\lambda_1}{\lambda_2} p_2(\lambda_1). \tag{31}$$

The left hand side of inequality (30) is derived from inequality (28) while the right side is derived from inequality (29) by setting $p_2(\ \lambda) = 1 - p_1(\ \lambda)$. Equation (31) is established in a similar fashion.

To prove inequalities (27), (28) and (29) we distinguish three cases:

1. $p_1(\lambda_2) = 1$: Then
$$E(\lambda_2) = R_1(\lambda_2) \leq \beta_2. \tag{32}$$
This implies that $p_1(\ \lambda_1) = 1$. To see this, note that if $p_1(\lambda_1) < 1$, then since $p_i(\lambda_2)$, $i = 1, 2$ satisfy the conditions of Proposition 1, we would have
$$E(\lambda_1) = R_1(\lambda_1 p_1(\lambda_1)) = R_2(\lambda_1 p_2(\lambda_1)) > \beta_2. \tag{33}$$
But because $\lambda_1 < \lambda_2$, it holds $R_1(\lambda_1 p_1(\lambda_1)) < R_1(\lambda_2) \leq \beta_2$. This inequality contradicts (33).

   Since for both arrival rates all the load is routed to S_1, all three inequalities are satisfied.

2. $p_1(\lambda_2) < 1, p_1(\lambda_1) = 1$: In this case,
$$E(\lambda_2) = R_1(\lambda_2 \times p_1(\lambda_2)) = R_2(\lambda_2 \times p_2(\lambda_2)) > \beta_2 \geq R_1(\lambda_1 \times p_1(\lambda_1)) = E(\lambda_1).$$
Hence inequality (27) is satisfied. Also, since $R_1(\lambda)$ is increasing, we conclude that $\lambda_1 < \lambda_2 \times p_1(\ \lambda_2)$. Inequality (29) is trivially satisfied.

3. $p_1(\lambda_1) < 1, p_1(\lambda_2) < 1$: Note first that it holds

$$p_i(\lambda_1) > 0,\ p_i(\lambda_2) > 0,\ i = 1, 2.$$

Indeed, $p_2(\lambda_j) > 0$ since $p_1(\lambda_j) < 1$. On the other hand, if $p_1(\lambda_j) = 0$ then we would have

$$R_1(\lambda_j p_1(\lambda_j)) = 0 < \beta_2 < R_2(\lambda_j p_2(\lambda_j),$$

which contradicts the assumptions that $p_i(\lambda_j)$, $i = 1, 2$, satisfies the conditions of Proposition 1.

Since

$$p_1(\lambda_1) + p_2(\lambda_1) = p_1(\lambda_2) + p_2(\lambda_2) = 1,$$

it must hold for $l = 1$ or 2,

$$p_l(\lambda_1) \leq p_l(\lambda_2).$$

Then, since $p_i(\lambda_1)$ and $p_i(\lambda_2)$ are positive, satisfy the conditions of Proposition 1, and $\lambda_1 < \lambda_2$, we have

$$E(\lambda_1) = R_l(\lambda_1 p_l(\lambda_1)) < R_l(\lambda_2 p_l(\lambda_2)) = E(\lambda_2).$$

Inequalities (28) and (29) are proved as in case 2, using the fact that $E(\lambda_1) < E(\lambda_2)$.

Proof of Proposition 2

The only if part is derived by observing that for routing probabilities $p_i(\lambda)$, $1 \leq i \leq N$, that induce finite average response times on each system we must have $\lambda p_i(\lambda) < \theta_i$, and therefore,

$$\lambda = \sum_{i=1}^{N} \lambda p_i(\lambda) < \sum_{i=1}^{N} \theta_i.$$

We use induction to prove the if part. That is, we will show by induction that for any N, if

$$0 < \lambda < \sum_{i=1}^{N} \theta_i, \tag{34}$$

then a unique routing probability vector $\mathbf{p}(\lambda)$ satisfying the conditions of Proposition 1 exists, and such that the induced $E(\lambda)$ satisfies the properties expressed in Proposition 2.

The statement is true for $N = 1$. In this case

$$p_1(\lambda) = 1 \text{ and } E(\lambda) = R_1(\lambda).$$

Now assume that the statement is true for $N = M$, and denote by $E_M(\lambda)$ the induced average overall response time when MMP is applied to M systems. Assume that we add a new system (i.e. S_{M+1}) with $\beta_{M+1} \geq \beta_M$. To complete the induction, given λ satisfying (34) we must find a routing probability vector for the set of systems $S_1, ..., S_M, S_{M+1}$, having the desired properties.

According to part a) of Proposition 2 we have that

$$\lim_{\lambda \searrow 0} E_M(\lambda) = \beta_1 \leq \beta_2 \leq \cdots \leq \beta_{M+1} = \lim_{\lambda \searrow 0} R_{M+1}(\lambda).$$

If $\beta_{M+1} > \beta_M$, then since $E_M(\lambda)$ is continuous and increases to infinity, there will be a rate A_{M+1} such that $E_M(A_{M+1}) = \beta_{M+1}$. If $\beta_{M+1} = \beta_M$ we define $A_{M+1} = A_M$. With this definition, and because $E_M(\lambda)$ satisfies condition b) of Proposition 2, we have that

$$0 \leq A_{M+1} < \sum_{i=1}^{M} \theta_i. \tag{35}$$

Let $\mathbf{p}_M(\lambda) = (p_1(\lambda), ..., p_M(\lambda))$ be the routing probability vector when MMP is applied to the M systems. When

$$0 < \lambda \leq A_{M+1}, \tag{36}$$

define

$$\mathbf{p}_{M+1}(\lambda) = (p_1(\lambda), ...p_M(\lambda), 0) \tag{37}$$

$$E_{M+1}(\lambda) = E_M(\lambda). \tag{38}$$

By the inductive hypothesis, $\mathbf{p}_{M+1}(\lambda)$ has the desired properties. Moreover, $E_{M+1}(\lambda)$ is continuous, strictly increasing in $(0, A_{M+1}]$ and satisfies part a) of Proposition 2. It remains to define $\mathbf{p}_M(\lambda)$ and $E_{M+1}(\lambda)$ when,

$$A_{M+1} < \lambda < \sum_{i=1}^{M+1} \theta_i. \tag{39}$$

We will show below that for λ satisfying (39) there is a unique number $q(\lambda)$ satisfying inequality

$$\max\left(0, \left(1 - \frac{\theta_{M+1}}{\lambda}\right)\right) < q(\lambda) < \min\left(1, \left(\sum_{i=1}^{M} \frac{\theta_i}{\lambda}\right)\right). \tag{40}$$

such that,

$$E_M(\lambda \times q(\lambda)) = R_{M+1}(\lambda \times (1 - q(\lambda))). \tag{41}$$

Assuming for the moment that such $q(\lambda)$ exists, we can proceed as follows.

Since by the inductive assumption $E_M(\lambda)$ satisfies Proposition 2 for $\lambda' = \lambda \times q(\lambda)$, we conclude that there is a set of probabilities $p_1', \cdots, p_M'$ such that

$$R_1(\lambda' \times p_1') = \cdots = R_M(\lambda' \times p_M') = E_M(\lambda') = R_{M+1}(\lambda \times (1 - q(\lambda))). \tag{42}$$

Now define for λ satisfying (39),

$$E_{M+1}(\lambda) = E_M(\lambda \times q(\lambda)). \tag{43}$$

$E_{M+1}(\lambda)$ is the required function, while the corresponding routing probability vector is:

$$\mathbf{p}_{M+1}(\lambda) = \left(p_1^{'}q(\lambda), \ldots, p_M^{'}q(\lambda), 1-q(\lambda)\right). \tag{44}$$

To see this, note first that $\mathbf{p}_{M+1}(\lambda)$ is a probability vector satisfying the conditions of Proposition 1. Hence, it remains to show that $E_{M+1}(\lambda)$ is strictly increasing, continuous for λ in $[A_{M+1}, \sum_{i=1}^{M+1}\theta_i)$ and satisfies part b) of Proposition 2.

From relations (36), (38), (39), (41) and (43) we observe that $E_{M+1}(\lambda)$ can be considered as the optimal solution applied to two systems: $\overline{S}_1$ with RTF $E_M(\lambda)$, and $\overline{S}_2$ with RTF $R_{M+1}(\lambda)$. For these two systems the routing probabilities are

$$\overline{p}_1(\lambda) = \begin{cases} 1 & 0 < \lambda \leq A_{M+1} \\ q(\lambda) & A_{M+1} < \lambda < \sum_{i=1}^{M+1}\theta_i \end{cases}, \ \overline{p}_2(\lambda) = 1 - \overline{p}_1(\lambda).$$

The fact that $E_{M+1}(\lambda)$ is strictly increasing follows directly from inequality (27). From inequality (30) it follows that $q(\lambda)$ is continuous. The continuity of $E_{M+1}(\lambda)$ follows from this fact and equation (43). To prove part b) observe that from inequalities (40) we have that

$$\left[\lambda \nearrow \sum_{i=1}^{M+1}\theta_i\right] \Rightarrow [\lambda \times (1 - q_M(\lambda)) \nearrow \theta_{M+1}],$$

and therefore

$$\lim_{\lambda \nearrow \sum_{i=1}^{M+1}\theta_i} E_{M+1}(\lambda) = \lim_{\lambda(1-q_M(\lambda)) \nearrow \theta_{M+1}} R_{M+1}\lambda(1 - q_M(\lambda)) = \infty.$$

It remains to prove the existence of a number $q(\lambda)$ satisfying (40) and (41). Let q satisfy (40) and consider the function

$$F(q) = E_M(\lambda \times q) - R_{M+1}(\lambda \times (1-q)).$$

$F(q)$ is finite, strictly increasing and continuous for q satisfying (40).
Also,

$$\lim_{q \searrow \max(0, 1-(\theta_{M+1}/\lambda))} F(q) = \begin{cases} -\infty & \text{if } \lambda \geq \theta_{M+1} \\ \beta_1 - R_{M+1}(\lambda) & \text{if } \lambda < \theta_{M+1} \end{cases}, \tag{45}$$

and

$$\lim_{q \nearrow \min\left(1, \left(\sum_{i=1}^{M}\frac{\theta_i}{\lambda}\right)\right)} F(q) = \begin{cases} \infty & \text{if } \lambda \geq (\theta_1 + \cdots + \theta_M) \\ E_M(\lambda) - \beta_{M+1} & \text{if } \lambda < (\theta_1 + \cdots + \theta_M) \end{cases}. \tag{46}$$

The limit in equation (45) is nonpositive. Also because of equation (39), we have that

$$E_M(\lambda) > E_M(A_{M+1}) = \beta_{M+1}.$$

Therefore, the limit in (46) is nonnegative. It follows that there is a unique root $q(\lambda)$ for $F(q)$ in the range specified by equation (40). Hence

$$F(q(\lambda)) = E_M(\lambda \times q(\lambda)) - R_{M+1}(\lambda \times (1 - q(\lambda)) = 0$$

as desired.

LEONIDAS GEORGIADIS received the Diploma degree in electrical engineering from Aristotle University, Thessaloniki, Greece, in 1979, and his M.S. and Ph.D degrees both in electrical engineering from the University of Connecticut, in 1981 and 1986 respectively. From 1981 to 1983 he was with the Greek army.

From 1986 to 1987 he was Research Assistant Professor at the University of Virginia, Charlottesville. In 1987 he joined IBM T. J. Watson Research Center, Yorktown Heights as a Research Staff Member. Since October 1995, he has been with the Telecommunications Department of Aristotle University, Thessaloniki, Greece. His interests are in the area of wireless networks, high speed networks, distributed systems, routing,scheduling, congestion control, modeling and performance analysis.

Prof. Georgiadis is a senior member of IEEE Communications Society. In 1992 he received the IBM Outstanding Innovation Award for his work on Goal-Oriented Workload Management for Multi-class systems.

CHRISTOS NIKOLAOU is Professor at the Dept. of Computer Science, Univ. of Crete, Greece. Currently he is Rector of the U. of Crete and Chairman of the Greek Ministry of Education, Committee on Informatics Policy for Education. He has been a Research Staff Member at IBM Thomas J. Watson Research Center, NY, USA, from 1981 to 1992. His research on resource allocation in high-performance transaction processing systems led to four US Patents, several IBM awards such as the IBM Outstanding Innovation Award for Scientific Contributions to Goal-Oriented Workload Management (1993) and refereed publications in scientific journals and conferences. He is chairman of the Executive Committee of ERCIM (European Research Consortium for Informatics and Mathematics, 95-98). He is also co-ordinator of the IST/FET Working Group iTrust (2002-2005), a forum for interdisciplinary investigation of trust as a means for establishing security and confidence in the global computing infrastructure, and as enabler for mutually beneficial interactions (www.iTrust.uoc.gr). Prof. Nikolaou is IEEE Senior Member.

ALEXANDER THOMASIAN has been a Professor at the Computer Science Dept. at the New Jersey Institute of Technology - NJIT since Sept. 2000. He spent fourteen years at IBM's T. J. Watson Research Center, one year of which was at the Almaden Research Center. After obtaining his PhD in Computer Science from UCLA he first became an Assistant Professor at Case Western University and then the University of Southern California. He is currently interested in the performance evaluation of computer systems, especially storage systems and high dimensional indexing methods. He has published several key papers on database concurrency control and the textbook: "Database Concurrency Control: Methods, Performance, and Analysis". He holds four patents, has received the IBM Innovation Award, and is the author of over 100 papers. A recent paper received the best paper award at the SPECTS 2003 Int'l Conf. held in Montreal, Canada. He was an editor of IEEE Trans. Parallel and Distributed Systems and has served on the program committees of numerous conferences. he has given tutorials at various conferences and offered short courses on performance evaluation. He is a Fellow of IEEE and a Member of ACM.