Comparison of Processor Allocation Policies for Parallel Systems

Rajesh K. Mansharamani
Mary K. Vernon

Technical Report #1194

November 1993

Comparison of Processor Allocation Policies for Parallel Systems *

Rajesh K. Mansharamani and Mary K. Vernon
mansha@cs.wisc.edu *vernon@cs.wisc.edu*

Computer Sciences Department
University of Wisconsin
1210 West Dayton Street
Madison, WI 53706.

December 9, 1993

Abstract

The increasing use of parallel systems has led to the development of a number of multiprogrammed processor allocation policies. This paper analyzes the following four policies that have previously been shown to have high performance under specific workloads: adaptive static partitioning (ASP), dynamic first-come-first-serve (FCFS), preemptive smallest available parallelism first (PSAPF), and spatial equipartitioning (EQS). The results in this paper are derived for a general workload model that includes general distribution of available job parallelism, controlled correlation between cumulative processing demand and available parallelism, general demand distribution per class of jobs in the correlation model, and general deterministic job execution rates that represent synchronization and communication overheads as well as load imbalance for parallel programs.

Under the assumption that jobs can dynamically and efficiently redistribute their work across the processors allocated to them previous interpolation approximations are used to estimate the mean response times of EQS and FCFS, and new interpolations are derived and validated for the mean response times of ASP and PSAPF. The interpolation approximations provide approximate mean response time formulas for each policy that directly yield key determinants of *relative* policy performance. The key determinants are used to delineate regions of the workload parameter space over which each of ASP, FCFS, EQS, and PSAPF performs best. The delineation provides a unification and generalization of previous results.

*This research was partially supported by the National Science Foundation under grants CCR-9024144 and CDA-9024618.

1 Introduction

The increasing use of parallel processor systems has led to the development of a number of multiprogrammed processor allocation policies. Many studies have compared the performance of specific processor allocation policies [7, 13, 14, 15, 16, 20, 21, 22, 25, 33, 34, 35, 37, 39, 41], which has led to a diverse set of results concerning relative policy performance over numerous specific regions of the workload parameter space.[1] For example,

- The Adaptive Static Partitioning (ASP) policy has been shown to have higher performance than several static allocation policies, under specific workload parameter values with *exponential per class total job processing requirements* (demands) [34].
- For a workload with independent and identically distributed *(i.i.d.) generalized exponential task service times* dynamic FCFS is shown to have higher performance than Round Robin Process and Processor Sharing when coefficient of variation of task service times is less than 4 [37].
- Under *exponential demands, no correlation between demand and parallelism, and linear speedups*, FCFS, EQuiallocation (EQ) policies, and Preemptive Smallest Available Parallelism First (PSAPF) perform almost the same [14, 18].
- PSAPF is optimal for a workload with *i.i.d. exponential task service times* [13], and also for a workload with *i.i.d. exponential job demands and linear speedups* [1], given that the scheduler has no information about job processing requirements.
- Under specific *hyperexponential demands and specific parallelism distributions*, PSAPF has been observed to have high performance under a workload with high correlation between demand and parallelism when coefficient of variation in job demand, C_D, is low to moderate[2]
- Under *specific hyperexponential demands and specific parallelism distributions* EQ has been observed to have high performance for both uncorrelated and highly correlated workloads when C_D is moderate to high [14].
- The spatial equipartitioning (EQS) policy is also shown to have high performance for *specific measurement workloads* [39, 20, 8] and a *particular workload that consists of a mixture of application types and has high* C_D [21, 22].

These results show particular policies to perform well over narrow regions of the parameter space but it is not clear whether or how the various results generalize. In particular, it is not clear which workload parameters determine relative policy performance or how the policies compare for say general distribution of job demand or of available parallelism, partial correlation between demand and parallelism, or speedup curves that range from highly sublinear to linear. The incomplete nature of previous studies and the lack of unifying results is due to several factors. First, analytic results in the literature are based on solution

[1] The performance metric throughout this paper is mean response time.

[2] By "low C_D" we mean C_D less than or equal to 1 and by "high C_D" we mean C_D in the range of 5 or more, as might be expected for general purpose workloads.

techniques that are applicable only to specific workload assumptions and thus the results are confined to narrow regions of the workload parameter space. Second, the numerical nature of solution techniques (e.g., solution of simultaneous equations, simulation, system measurement) yields no direct insight about how the policies perform outside the assumed workloads. Third, for each given set of specific workload assumptions, different subsets of possible high-performance policies have been compared.

This paper analyzes and compares the performance of four policies that have been shown to have high performance for specific workloads, viz. ASP, EQS, FCFS, and PSAPF, over a general workload model that we believe captures the essential features of parallel applications. The workload model used in this study is defined in [18, 19] and includes general distribution of *available* job parallelism[3], controlled correlation between total job processing requirement (demand) and parallelism, general distribution of demand for each class of jobs in the correlation model, and general deterministic job execution rates that represent synchronization and communication overheads as well as load imbalance for parallel programs.

Under the assumption that jobs can dynamically and efficiently redistribute their work across the processors allocated to them, we obtain mean response time estimates for each policy primarily using interpolation approximations, which were introduced in [18] and refined for the EQS policy in [19]. The approximate mean response time formulas that follow from the interpolations readily identify workload parameters that are key *determinants* of relative policy performance, and are used to evaluate policy performance as a function of these key parameters. Using the key determinants, interpolation approximations, and simulation in a few cases, we delineate regions of the model parameter space over which each policy performs best and thus generalize and unify results in the literature for the relative performance of ASP, EQS, FCFS, and PSAPF. A complete design space exploration such as this is not generally possible if policy performance is evaluated using numerical analysis, simulation, or measurement because in these techniques the functional dependence of policy performance on key workload parameters is not apparent.

The remainder of this paper is organized as follows. Section 2 presents the system model and defines the ASP, EQS, FCFS, and PSAPF processor allocation policies. Section 3 reviews and develops interpolation approximations for estimating the mean response times of each of these four policies, and Section 4 validates the new interpolation approximations. Section 5 uses the approximate mean response time formulas to compare policy performance, and finally Section 6 summarizes the conclusions of this study. Analytic proofs

[3]The available job parallelism of a job is the number of processors the system scheduler believes the job can productively use.

and derivations are given in the appendix.

2 System Model

We consider an open system model with P identical processors and a central job queue as shown in Figure 1. The centralized queueing model is a conceptual model; actual implementations of the scheduling policy may in general allow for distributed queue access. We assume zero scheduling and preemption overhead, with the understanding that the actual implementation of a particular scheduling policy will include limits on preemption rates (i.e., delayed preemptions) so as to reduce overhead to a small fraction of the productive execution on the processors.

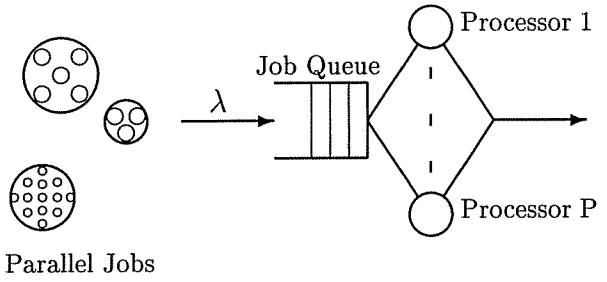


Figure 1: Open System Model

Below we define the processor allocation policies of interest (Section 2.1) and the workload model as it was defined in [18, 19] (Section 2.2). From [19] we also review constraints that exist among workload parameters (Section 2.3), and finally, we define the notation used throughout the remainder of the paper (Section 2.4).

2.1 Processor Allocation Policies

As stated in Section 1 we compare the following processor allocation policies in this paper: ASP, FCFS, EQS, and PSAPF. The ASP policy has been identified in previous work as a high performance static allocation policy [33]. The FCFS policy is very simple and has been shown to have high performance for specific workloads [37]. Equiallocation policies have been shown to have high performance under various workloads [39, 14, 20, 8, 33, 21, 22, 18]. The EQS policy is an idealized spatial equiallocation policy. Finally,

we examine the PSAPF policy that was proposed in [15] and shown to have high performance for specific workloads in [15, 13, 1].

Each of the policies is defined in the context of a global or central job queue. The four policies are defined in terms of the processing power that they allocate to jobs in the queue, and not in terms of the allocation of processors to individual tasks within a job. All four policies make use of the *available parallelism* in the jobs to decide how many processors to allocate to each job in the system. We define the available parallelism of a job to be the number of processors the scheduler believes the job can make productive use of.

ASP Adaptive Static Partitioning. When a job arrives it is allocated the lesser of the number of idle processors in the system and its available parallelism. If no free processors are available, the job queues behind waiting jobs, if any. When a job completes, the released processors are allocated one at a time to the waiting jobs in round robin order (starting from the first waiting job), under the constraint that no job is allocated more processors than its available parallelism. For example, if there are five jobs waiting when a fully parallel job completes in a 100-processor system and the available parallelism per job is (50, 25, 100, 10, 10), then the allocation of processing power is (28, 25, 27, 10, 10). This policy was first defined in [33] and its performance has been studied under several workload assumptions [34, 29, 21, 22].

FCFS The FCFS policy[4] allocates processors to jobs on a first-come-first-serve basis. Each job is allocated processors as they become available up to a maximum of its available parallelism. Processors released by a departing job are first allocated to the job in service (if any) whose allocation is less than its available parallelism and then to jobs waiting for service. Allocation of processing power for the five jobs in the above example is (50, 25, 25, 0, 0). This policy has been studied under different workload assumptions in previous literature [26, 15, 23, 14, 37, 13].

EQ Dynamic equiallocation (EQ) policies allocate an equal fraction of processing power to each job in the system unless a job has smaller available parallelism than the equiallocation value, in which case each such job is allocated as many processors as its available parallelism, and the equiallocation value is recursively recomputed for the remaining jobs. Allocation of processing power for the above example is (27.5, 25, 27.5, 10, 10). Reallocation of power can occur on job arrivals, job departures, and changes in a job's available parallelism.

EQS The Spatial EQuiallocation policy is an EQ policy in which processing power is allocated spatially for integral allocation and temporally for fractional allocation. For example, if a job is to receive an allocation of 27.5 units of processing power, then it is allocated 27 processors and it receives an additional 0.5 units of processing power by time sharing an additional processor (i.e., the job alternately executes on 27 and 28 processors). Ignoring variations in implementation details, the EQS policy was first defined in [39].

PSAPF Preemptive Smallest Available Parallelism First. The central job queue is a preemptive queue that is ordered in ascending order of available job parallelism. Jobs with the same available parallelism are served in first-come-first-serve order. As in the FCFS policy each job is allocated processors as they become available (or preempted) up to a maximum of its available parallelism, and processors released by a departing job are first allocated to the job in service (if any) whose allocation is less than its available parallelism and then to the jobs waiting for service. Processor allocation for the

[4]The FCFS policy is defined for the case that available parallelism is fixed throughout the life of a job, as assumed in the workload model in Section 2.2. There exist extensions to the policy for the case where the available parallelism of the job changes during its lifetime and the system scheduler can detect and react to the changes.

above example is (50, 25, 5, 10, 10). Processor allocation to jobs can change upon job arrivals, job departures, and changes in job parallelism. This policy was proposed in [15] and also studied in [13, 14] under specific workloads.

2.2 Workload Model

The goal is to have a simple workload model that is broadly applicable, uses a small number of parameters to characterize the essential features of parallel workloads with respect to scheduling disciplines, and is easy to analyze. To achieve broad applicability, few restrictions are made on the distribution of important system parameters, such as job parallelism and total service demand. To keep the parameter space simple and to facilitate ease of analysis, a simple characterization of job execution rates and correlation between demand and parallelism is assumed.

Jobs arrive to the system according to a Poisson process with rate λ as shown in Figure 1. All jobs are assumed to be statistically identical. Each job is characterized by the following variables.

(1) Total service demand (execution time on one processor) D,

(2) Available parallelism $N \in \{1, 2, \ldots, P\}$,

(3) Execution rate function (ERF) $E : [0, P] \rightarrow [0, N]$, which is nondecreasing and has the following properties:

$$E(x) \begin{cases} \leq x, & 0 \leq x \leq N, \\ = E(N), & N < x \leq P. \end{cases}$$

(4) Correlation coefficient $r \in [0, 1]$ between available parallelism N and the random variable for mean demand of a job with available parallelism N.

The system operates as follows. Upon arrival each job joins the central job queue. At each time, $t \geq 0$, the P processors are allocated to jobs present in the queue according to the processor allocation policy Ψ. If $a(t)$ processors (possibly fractional) are allocated to a job at time t, then its demand is satisfied at rate $E(a(t))$. In other words, $E(k)$ is the *speedup* of the job if the job is allocated k processors throughout its execution, and if the allocation can vary $E(x)$ is also assumed to be the *instantaneous rate* at which the job executes whenever it is allocated x processors. The job leaves the system upon completion of its total demand, D. The available parallelism, N, of a job is the number of processors the system scheduler believes

the job can productively use. The workload model assumes that N is an upper bound on the actual number of processors, m, the job can productively use (i.e., by definition, $E(x) = E(N)$ for $N < x \leq P$, and if m is less than N then $E(j) = E(m), m < j \leq N$.)

The following is assumed about N, E, D, and r.

- N has a general (bounded) distribution with mean $\overline{N}$, coefficient of variation[5] C_N, and probability mass function $\underline{p} = (p_1, \ldots, p_P)$, where $p_k = \Pr[N = k]$, $k = 1, \ldots, P$.

- E is derived from a *deterministic* function γ, that is nondecreasing and is such that $\gamma(x) = x$ for $0 \leq x \leq 1$, and $\gamma(x) \leq x$ for $1 < x \leq P$.

  For a job with available parallelism N, $E(N) = \gamma(N)$. When fewer than N processors are allocated to the job, the execution rate E depends on more detailed characteristics of the applications. In this paper, we assume that the work for a job can be dynamically redistributed across the number of processors allocated to it such that it executes as if it had available parallelism equal to the processor allocation, i.e., $E(j) = \gamma(j)$, for $1 \leq j < N$. This could be appropriate for applications based on the work queue model, or in some cases where the processes of a job are timeshared on the allocated processors. In cases where the allocated processing power, x, is nonintegral we use a linear interpolation between $\gamma(\lfloor x \rfloor)$ and $\gamma(\lceil x \rceil)$ to compute $E(x)$.

  Note that other assumptions about job execution rate on fewer than N processors are possible. For example, one might assume that the parallelism overhead is about the same on fewer processors as on N processors, i.e., $E(j) = \frac{j}{N}\gamma(N)$ for $1 \leq j < N$, which could represent a system with jobs that have fixed parallelism in which overhead is primarily due to message passing software and processing load is balanced across the processors, e.g., through cyclic rotation of processes. As another example, if communication overheads are fixed for a given available parallelism but the load is only balanced when j evenly divides N, then $E(j) = \frac{1}{\lceil N/j \rceil}\gamma(N)$, for $1 \leq j < N$.

  As shown by the above examples, for given assumptions about the application characteristics, the function γ *determines* the ERF E. Thus γ will be called the execution rate determinant (ERD) of the workload in the remainder of this paper. The ERD γ is said to be *linear* if $\gamma(x) = x$, for all $0 \leq x \leq P$.

- The mean demand of a job with available parallelism N is either independent of N or linearly correlated with N. We assume the following model from [19] in which the mean demand of a job with available parallelism N is given by

$$\Delta_N = \begin{cases} \overline{D}, & \text{with probability } 1 - r^2, \\ c\,N, & \text{with probability } r^2. \end{cases}$$

  In the first case the demand is drawn from a general distribution, $\mathcal{F}_D^u$, with mean $\overline{D}$ and coefficient of variation C_v, where C_v is a fixed constant independent of N. In the second case, the demand is stochastically equal to a demand that is drawn from the same distribution and then scaled by the factor $\frac{c\,N}{\overline{D}}$. In the latter case the mean demand is $c\,N$ as required, and the coefficient of variation of is equal to C_v, which does not depend on N. It is easy to see after unconditioning on N that $c = \overline{D}/\overline{N}$. Thus, the workload correlation is captured by the single parameter r, which is shown in [19] to be equal to the correlation coefficient between Δ_N and N, i.e.,

$$r = Corr(\Delta_N, N) \equiv \frac{E[\Delta_N N] - E[\Delta_N]\,E[N]}{\sigma_{\Delta_N}\sigma_N}, \quad \sigma_{\Delta_N}, \sigma_N \neq 0. \tag{1}$$

[5] The coefficient of variation of a random variable is the ratio of the standard deviation to the mean.

The service time of a job on N processors is denoted by the random variable $S = D/\gamma(N)$, with mean denoted by $\overline{S}$. Under the above workload assumptions the mean service time under arbitrary $r \in [0,1]$ is given by

$$\begin{aligned} \overline{S} = E\left[\frac{D}{\gamma(N)}\right] &= (1-r^2)\overline{D}E\left[\frac{1}{\gamma(N)}\right] + r^2\frac{\overline{D}}{\overline{N}}E\left[\frac{N}{\gamma(N)}\right] \\ &= (1-r^2)\,\overline{S}(r=0) + r^2\overline{S}(r=1). \end{aligned} \tag{2}$$

The workload model defined above contains four simplifications each of which represents a trade-off between analytic tractability and the simplicity of the parameter space on the one hand, and generality of the model on the other hand. The first is the assumption of constant available parallelism per job, the second is the assumption of a fixed execution rate, $E(k)$, whenever the job is allocated k processors, the third is the assumption of the same deterministic execution rate function γ for all jobs, and the fourth is the specific type of correlation model assumed. The implications of these assumptions are discussed further in [19]. An important point is that since the purpose of the model is to analyze *scheduling policy* performance, as opposed to obtaining mean response time for the applications, assumptions that represent key workload characteristics while keeping the model tractable and the parameter space simple, are acceptable even when they do not precisely describe the behavior of individual applications. For example, the simple correlation model may not precisely model a particular realistic workload, but one can use it to vary the degree of correlation between mean demand and available parallelism and thereby study the impact of correlation on policy performance.

2.3 Constraints on the Model Parameters

Workload parameters of immediate interest to us are mean and coefficient of variation in demand, i.e., $\overline{D}$ and C_D, mean and coefficient of variation of available parallelism, i.e., $\overline{N}$ and C_N, correlation coefficient r, execution rate function γ, and mean service time $\overline{S}$. The parameters $\overline{D}$, $\overline{N}$, γ, and r can vary freely within their feasible ranges (e.g., $0 \leq \overline{D} < \infty$, $1 \leq \overline{N} \leq P$, or $0 \leq r \leq 1$), and thus are the free parameters of the model. Below, relationships that constrain the other parameters of interest, i.e., C_D, C_N, and $\overline{S}$, are reviewed. The relationships define the parameter space for the policy comparisons.

The coefficient of variation, C_D, in demand (after unconditioning on N) can vary freely between 0 and ∞ only when D and N are independent, i.e., $r = 0$. For $r > 0$, it can be verified that that C_D depends on

C_v, r, and C_N as follows:

$$C_D^2 = (1 + C_v^2)(1 + r^2 C_N^2) - 1. \tag{3}$$

Since N is bounded above by P, it follows that C_N cannot be unbounded. For a given $\overline{N}$, the following constraints on C_N are derived in [19],

$$0 \leq C_N \leq \sqrt{\frac{\overline{N}(P+1) - P}{\overline{N}^2} - 1}\ . \tag{4}$$

The lower bound is attained when N is constant and integer-valued for all jobs, i.e., $N = k$, where $k \in \{1, \ldots, P\}$. The upper bound is attained when N has a two-point p.m.f. with nonzero mass only at 1 and P.

The following constraints on $\overline{S}$ are derived in [19]. When $r = 0$ and γ is concave[6], $\overline{S}$ is minimum when C_N is minimum and $\overline{S}$ is maximum when C_N is maximum. When $r = 1$, γ is concave, and $N/\gamma(N)$ is concave, $\overline{S}$ is maximum when C_N is minimum and $\overline{S}$ is minimum when C_N is maximum. For concave γ and $N/\gamma(N)$, $\overline{S}$ decreases with workload correlation r. (Note that $N/\gamma(N)$ is concave for the concave ERD considered in the experiments in this paper.)

2.4 Notation

Table 1 summarizes the notation for the system parameters and variables. Under the implicit assumption of Poisson arrivals and system of P processors we use the following notation to characterize specific system workloads.

$$(\Psi,\ \lambda,\ \mathcal{F}_N,\ \mathcal{F}_D^u,\ r,\ \gamma,\ E(j)),$$

Ψ = processor allocation policy

λ = job arrival rate

$\mathcal{F}_N$ = distribution of N, e.g., $N = P$, Uniform(1,P)

$\mathcal{F}_D^u$ = distribution of demand for jobs with mean demand independent of parallelism e.g., exp(μ)

r = correlation coefficient

[6] A function $f : (a, b) \rightarrow IR$ is concave if $f(\alpha x + (1-\alpha)y) \geq \alpha f(x) + (1-\alpha)f(y)$, for all $x, y \in (a, b)$ and $\alpha \in (0, 1)$. Conversely, f is convex if $f(\alpha x + (1-\alpha)y) \leq \alpha f(x) + (1-\alpha)f(y)$ [28]. Informally, a function is concave if the straight line joining any two points of the function lies on or below all function values between the two points, and is convex if the line lies on or above the function values.

γ = execution rate determinant. By default we assume that γ is a general nondecreasing function. To specify the linear ERD we use the notation γ^l.

$E(j)$ = job execution rate on $j < N$ processors , e.g., $E(j) = \gamma(j)$ in the case of jobs that can dynamically and efficiently redistribute their work.

To indicate a general distribution of demand or available parallelism, general ERD, or arbitrary value of r between 0 and 1, we simply leave the notation as $\mathcal{F}_D^u$, $\mathcal{F}_N$, γ, or r, respectively.

Table 1: System Notation

P	Number of processors in the system
λ	Arrival rate of jobs
D	Total job demand
$\mathcal{F}_D^u$	Distribution of demand for "uncorrelated" jobs
$\overline{D}$	Overall mean job demand
C_D	Overall coefficient of variation of demand
ρ	Offered load $\lambda\overline{D}/P$
N	Available job parallelism
$\mathcal{F}_N$	Distribution of available parallelism
p_k	Probability$[N = k]$, $k = 1, \ldots, P$
$\underline{p}$	$(p_1, p_2, \ldots, p_P)$
$\overline{N}$	Average available parallelism
C_N	Coefficient of variation of available parallelism
r	Measure of workload correlation as defined in (1)
γ	Execution rate determinant (ERD) of the workload
γ^l	Linear execution rate function
$\overline{S}$	Mean job service time
S_n	Normalized mean service time $\overline{S}/\overline{D}$
$\overline{R}_\Psi$	Mean response time of policy Ψ
$M/G/1_P$	An $M/G/1$ system with a server of power P

2.5 Workloads for Numerical Experiments

The following distributions of N and functions γ are used to validate mean response time approximations in Section 4 and to experimentally compare specific policies in Section 5.

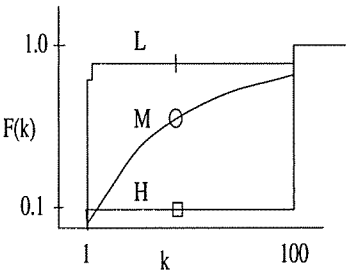
- The bounded-geometric distribution of N with parameters P_{max} and p (see [14, 12]):

$$N = \begin{cases} P, & \text{with probability } P_{max}, \\ \min(X, P), & \text{with probability } 1 - P_{max}, \end{cases} \quad \text{where } X = \text{Geometric}(p).$$

It can be verified that across all bounded-geometric distributions with $\overline{N}$ the same, C_N is maximum when $p = 1$ and is minimum when $P_{max} = 0$ [17]. We refer to these workloads as **high** C_N and **low** C_N workloads, respectively. The specific bounded-geometric distributions in Table 2, more details of which are given in [18], are used in the policy comparisons.

Table 2: Three Bounded-Geometric Distributions for N
P=100

Symbol	Parallelism	P_{max}	p	$\overline{N}$	C_N	CDF of N
H	High	0.9	1.0	90.10	0.33	
M	Moderate	0.1	1/(0.4P)	43.14	0.80	
L	Low	0.1	0.9	11.00	2.70	



- To evaluate the impact of sublinear ERFs, the following parametric ERD will be considered, which is derived from an execution signature in [6].

$$\gamma(k) = \frac{(1+\beta)k}{k+\beta}, \quad k = 1, 2, \ldots. \tag{5}$$

At $\beta = 0$ we get the flat ERD $\gamma(k) \equiv 1$. By increasing β we obtain ERDs that are closer to linear as shown in Figure 2 until we obtain the linear ERD when $\beta = \infty$. We note that $\beta = 100$ represents a considerably sublinear ERD, whereas $\beta = 500$ is fairly close to linear.

For $r = 0$ and the ERD (5),

$$S_n(\gamma) = E\left[\frac{1}{\gamma(N)}\right] = E\left[\frac{N+\beta}{(1+\beta)N}\right] = \frac{1}{1+\beta}\left(1 + \beta E\left[\frac{1}{N}\right]\right) = \frac{1}{1+\beta}[1 + \beta S_n(\gamma^l)],$$

or equivalently,

$$S_n(\gamma^l) = \frac{1}{\beta}\left[(1+\beta)S_n(\gamma) - 1\right]. \tag{6}$$

Equation (6) will prove useful in exploring the design space of relative policy performance of EQS and ASP in Section 5.1.

3 Approximate Analysis

The goal is to solve for the performance of the FCFS, PSAPF, and ASP policies under general distributions of demand $\mathcal{F}_D^u$ and available parallelism $\mathcal{F}_N$, arbitrary correlation coefficient r, and any general ERD γ, as

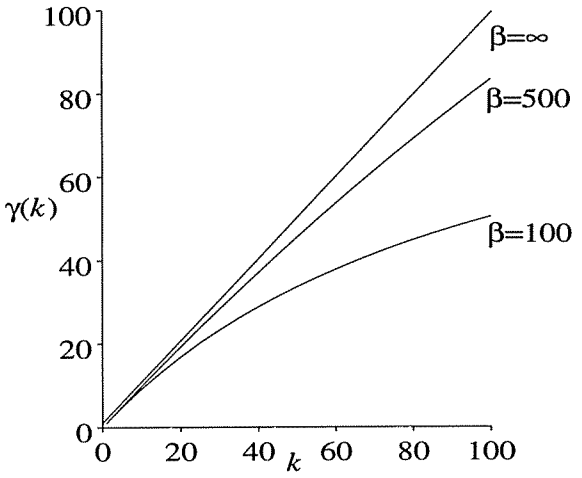


Figure 2: ERD $\gamma(k) = (1+\beta)k/(k+\beta)$

was done for the EQS policy in [19]. However, the FCFS, ASP, and PSAPF policies are difficult to analyze under such completely general workload assumptions and therefore suitable restrictions are made below for the sake of analytic tractability. Fortunately, the restrictions do not limit the applicability of the policy comparison results, because as will be shown, one can extrapolate from the comparisons under the restricted assumptions to the general case.

Interpolation approximations are derived by obtaining accurate estimates of mean response time (either exact or approximate) at extreme values of system parameters and then interpolating among the end-points. In this section we review interpolation approximations from [18] for $\overline{R}_{FCFS}$ under (λ, $\mathcal{F}_N$, $\mathcal{F}_D$, $r = 0$, γ^l, $E(j) = \gamma(j)$) and from [19] for $\overline{R}_{EQS}$ under (λ, $\mathcal{F}_N$, $\mathcal{F}_D^u$, r, γ, $E(j) = \gamma(j)$). New interpolation approximations or reductions are derived for the following policies and workloads assuming an arrival rate of λ and $E(j) = \gamma(j)$:

FCFS:	$(N = k,\ \mathcal{F}_D,\ r = 0,\ \gamma)$,
ASP:	$(\mathcal{F}_N,\ \exp(1/\overline{D}),\ r = 0,\ \gamma^l)$,
ASP:	$(N = P,\ \exp(1/\overline{D}),\ r = 0,\ \gamma^l)$,
PSAPF:	$(N = k,\ \mathcal{F}_D,\ r = 0,\ \gamma)$,
PSAPF:	$(\mathcal{F}_N,\ \mathcal{F}_D,\ r = 0,\ \gamma^l)$, and
PSAPF:	$(\mathcal{F}_N,\ \mathcal{F}_D^u,\ r > 0,\ \gamma^l)$.

As will be shown in Section 5, the restrictive assumptions $(r = 0, \gamma^l)$ for FCFS under general $\mathcal{F}_N$ provides

the best case performance of FCFS relative to PSAPF. Likewise, the assumptions $(\exp(1/\overline{D}), r = 0, \gamma^l)$ are favorable for ASP relative to EQS, and the assumption (γ^l) will be shown favorable for PSAPF relative to EQS. Section 5 will show that conclusions from these "favorable" comparisons can be extrapolated to other regions of the general workload parameter space. The mean response time estimates for FCFS and PSAPF under $(N = k, r = 0, \gamma)$ provide insight about how the performance of each of these policies behaves with respect to ERF sublinearity.

Sections 3.1 and 3.2 provide the approximations for $\overline{R}_{FCFS}$ and $\overline{R}_{EQS}$, respectively. We then develop the (interpolation) approximations for $\overline{R}_{ASP}$ (Section 3.3), and for $\overline{R}_{PSAPF}$ under no correlation between mean demand and available parallelism (Section 3.4) and under full correlation (Section 3.5). The new approximations for $\overline{R}_{ASP}$ and $\overline{R}_{PSAPF}$ are validated in Section 4.

3.1 FCFS

First an accurate interpolation approximation for $\overline{R}_{FCFS}$ from [18] is reviewed that holds for $(\lambda, \mathcal{F}_N, \mathcal{F}_D, r = 0, \gamma^l, E(j) = \gamma(j))$. We then develop a new approximation for $\overline{R}_{FCFS}$ under constant available parallelism and general γ.

3.1.1 Analysis under General N: $r = 0$ and γ^l

For the system $(FCFS, \lambda, \mathcal{F}_N, \mathcal{F}_D, r = 0, \gamma^l, E(j) = \gamma(j))$ the following interpolation approximation on the pmf of N, $\underline{p}$, is an accurate estimate for $\overline{R}_{FCFS}$ [18]:

$$\begin{aligned} \overline{R}_{FCFS}(\mathcal{F}_N, r = 0) &\approx \sum_{k=1}^{P} p_k \overline{R}_{FCFS}(N = k, r = 0), \quad \text{under } (\lambda, \cdot, \mathcal{F}_D^u, \cdot, \gamma^l, E(j) = \gamma(j)) \\ &= \overline{S} + \frac{E\left[\rho^{\sqrt{2(\frac{P}{N}+1)}}\right]}{1-\rho} \left(\frac{1+C_D^2}{2\lambda}\right), \end{aligned} \tag{7}$$

where the solution for $\overline{R}_{FCFS}(N = k, r = 0)$ is derived by reducing the system to the M/G/c queue, under similar reasoning to the reduction under the more general assumptions of $(N = k, r = 0, \gamma)$, given next.

3.1.2 Analysis under Constant N

Let $\Gamma_{FCFS,k} = (FCFS, \lambda, N = k, \mathcal{F}_D, r = 0, \gamma, E(j))$. In [18] mean response time estimates were provided for this system under the assumption that $\gamma = \gamma^l$. The extension to general nondecreasing γ is straightforward

and uses the following reduction.

First consider the case where k evenly divides P. A job arriving at an empty system is allocated k processors. Subsequent jobs that arrive are also allocated k processors unless all processors are occupied. When a job departs it releases all k of its processors as a single unit. The first job waiting in the queue (if any) thus obtains all k processors released by the departing job, and so on. Since processors are allocated and released in units of size k, the system $\Gamma_{FCFS,k}$ behaves like a system with $c = P/k$ processors in which each job has one task with service requirement $S = D/\gamma(k)$. That is, under $(\lambda, N = k, \mathcal{F}_D, r = 0, \gamma, E(j))$

$$\overline{R}_{FCFS}(N = k, r = 0) \;=\; \overline{R}_{M/G/c}, \quad c = P/k, \quad P \bmod k = 0.$$

To compute $\overline{R}_{M/G/c}$ we use the following approximation which is derived using Sakasegawa's approximation [30] for the mean number in a GI/G/c queue:

$$\overline{R}_{M/G/c} \approx \overline{S} + \frac{\nu\sqrt{2(c+1)}}{1-\nu}\left(\frac{1+C_S^2}{2\lambda}\right), \quad \text{where} \quad \nu = \frac{\lambda\overline{S}}{c},$$

C_S being the coefficient of variation in job service time, S. Using $S = D/\gamma(k)$, we obtain $C_S^2 = C_D^2$ and thus

$$\overline{R}_{FCFS}(N = k, r = 0) \approx \frac{\overline{D}}{\gamma(k)} + \frac{\nu\sqrt{2(P/k+1)}}{1-\nu}\left(\frac{1+C_D^2}{2\lambda}\right), \quad \text{under } (\lambda, \cdot, \mathcal{F}_D^u, \cdot, \gamma, E(j)) \tag{8}$$

where $\nu = \dfrac{\lambda\overline{D}}{P} \cdot \dfrac{k}{\gamma(k)}$. Since (8) can also be computed when k does not evenly divide P, it can be used as an approximation for $\overline{R}_{FCFS}(N = k)$ for all $k = 1, 2, \ldots, P$.

It is tempting to believe that the interpolation on $\underline{p}$ using estimates from (8) as the interpolation endpoints can be used to approximate $\overline{R}_{FCFS}$ for a sublinear ERD γ. Validations so far have shown that this approximation is accurate for low values of ρ, but that the accuracy degrades with ρ and at high load the accuracy can be quite poor even when $C_D = 1$ (a case under which the approximation is very accurate for the linear ERD). With some "fine tuning" it may be possible to obtain an accurate estimator by this approach, however this is not pursued further in this paper.

Note that both approximations (7) and (8) show that $\overline{R}_{FCFS}$ increases linearly with C_D^2.

3.2 EQS

Below is a summary of the mean response time solutions from [19] for the system $(EQS, \lambda, \cdot, \mathcal{F}_D^u, r, \gamma, E(j) = \gamma(j))$. First an accurate approximation under general $\mathcal{F}_N$ (Section 3.2.1) and second the exact solution under constant N (Section 3.2.2), are reviewed.

3.2.1 Approximation for general N

The results in [19] show that the normalized mean service time $S_n \equiv \overline{S}/\overline{D}$ is the key parallelism determinant of EQS performance under the workload $(\lambda, \mathcal{F}_N, \mathcal{F}_D^u, r, \gamma)$ and that $\overline{R}_{EQS}$ increases linearly with S_n. Thus, the following linear interpolation on S_n is an accurate estimate of $\overline{R}_{EQS}$.

$$\begin{aligned}\overline{R}_{EQS}(\mathcal{F}_N, r) \approx & \left(\frac{S_n - 1/\gamma(P)}{1 - 1/\gamma(P)}\right)\overline{R}_{EQ}(N = 1, r = 0) + \\ & \left(\frac{1 - S_n}{1 - 1/\gamma(P)}\right)\overline{R}_{EQ}(N = P, r = 0), \text{ under } (\lambda, \cdot, \mathcal{F}_D^u, \cdot, \gamma, E(j) = \gamma(j))\end{aligned} \tag{9}$$

The mean response times $\overline{R}_{EQS}(N = 1, r = 0)$ and $\overline{R}_{EQS}(N = P, r = 0)$ are special cases of the exact solution in [19] for $\overline{R}_{EQS}(N = k, r = 0)$, $k = 1, 2, \ldots, P$, which is reviewed below in Section 3.2.2.

Interpolation (9) can be rewritten by expressing S_n as $(1 - r^2)S_n(r = 0) + r^2 S_n(r = 1)$ (see (2)), which results in,

$$\overline{R}_{EQS}(r) \approx (1 - r^2)\overline{R}_{EQS}(r = 0) + r^2\overline{R}_{EQS}(r = 1), \quad \text{under } (\lambda, \mathcal{F}_N, \mathcal{F}_D^u, \cdot, \gamma, E(j) = \gamma(j)), \tag{10}$$

as shown in [19]. $\overline{R}_{EQS}(r = 0)$ and $\overline{R}_{EQS}(r = 1)$ can be directly derived from (9).

3.2.2 Exact solution for $N = k$

The derivation for $\overline{R}_{EQS}(N = k, r = 0)$ is based on the property that when $N = k$ the EQS system reduces to a *symmetric queue* [9], and it is shown in [19] that under the workload assumptions (λ, $N = k$, $\mathcal{F}_D$, $r = 0$, γ, $E(j) = \gamma(j)$)

$$\begin{aligned}\overline{R}_{EQS}(N = k, r = 0) = & \frac{b}{\lambda}\left\{\sum_{i=1}^{P}\frac{(P\rho)^i}{(i-1)!\,E(k)^{\min(i,m)}\prod_{j=m+1}^{i}E(P/j)} + \right. \\ & \left.\frac{(P\rho)^P}{P!\,E(k)^m\prod_{j=m+1}^{P}E(P/j)}\frac{\rho}{1-\rho}\left(\frac{1}{1-\rho} + P\right)\right\}, \quad k = 1, 2, \ldots, P,\end{aligned}$$

where $m = \lfloor P/k \rfloor$, $\rho = \lambda \overline{D}/P$, and

$$b = \left[1 + \sum_{i=1}^{P} \frac{(P\rho)^i}{i!\, E(k)^{\min(i,m)} \prod_{j=m+1}^{i} E(P/j)} + \frac{(P\rho)^P}{P!\, E(k)^m \prod_{j=m+1}^{P} E(P/j)} \frac{\rho}{1-\rho} \right]^{-1}.$$

Note that the above expression for $\overline{R}_{EQS}$ as well as the expression in (9) are independent of C_D.

3.3 ASP

In this section an approximation is developed for $\overline{R}_{ASP}$ under $(\lambda, \mathcal{F}_N, \exp(1/\overline{D}), r = 0, \gamma^l, E(j) = \gamma(j))$. Setia and Tripathi [33] derive an exact solution for $\overline{R}_{ASP}$ under exponential per class job demands and general job execution rates, which is based on matrix-geometric analysis [27, 24]. Two drawbacks of this exact analysis are that the underlying state space grows exponentially in the number of processors (making the analysis computationally prohibitive even for systems with 20 processors) and that the analysis does not yield direct insight into the dependence of $\overline{R}_{ASP}$ on workload parameters.

In contrast, a closed form approximation is developed below for $\overline{R}_{ASP}$ under the restrictive assumptions of linear execution rates and exponential demands. The assumption of linear execution rates yields estimates of the best possible performance of ASP (i.e., under no synchronization and communication overheads). The exponential demand assumption should also result in lower estimates for $\overline{R}_{ASP}$ than for workloads with high C_D, since ASP is a static allocation policy. This is discussed further in Section 5.

Section 3.3.1 presents an interpolation approximation under general $\mathcal{F}_N$ and Section 3.3.2 derives reductions and interpolation approximations for the extreme cases of constant N, i.e., $N = 1$ and $N = P$.

3.3.1 Analysis under General N: $\exp(1/\overline{D})$, $r = 0$, γ^l

To derive an approximation for $\overline{R}_{ASP}$ we note from Section 2 that at each allocation point ASP divides processors equally among waiting jobs (with no fewer than one processor per job). This resemblance to the EQS policy suggests using the same form of interpolation for $\overline{R}_{ASP}$ as we used for $\overline{R}_{EQS}$ in (9), that is, an interpolation on S_n. Thus we have the following interpolation approximation on S_n for the mean response time of $(ASP, \lambda, \mathcal{F}_N, \exp(1/\overline{D}), r = 0, \gamma^l, E(j) = \gamma(j))$.

$$\overline{R}_{ASP}(\mathcal{F}_N) \approx \left(\frac{S_n - 1/P}{1 - 1/P} \right) \overline{R}_{ASP}(N = 1) +$$

$$\left(\frac{1-S_n}{1-1/P}\right)\overline{R}_{ASP}(N=P), \quad \text{under } (\lambda,\cdot,\exp(1/\overline{D}),r=0,\gamma^l,E(j)=\gamma(j)). \quad (11)$$

Solutions for $\overline{R}_{ASP}(N=1)$ and $\overline{R}_{ASP}(N=P)$ are given next.

3.3.2 Analysis for $N=1$ and $N=P$: $\exp(1/\overline{D})$, γ^l

When $N=1$, ASP is the same as FCFS. Therefore for exponential job demands, $\overline{R}_{ASP}(N=1)$ is simply the mean response time in an $M/M/P$ queue, i.e.,

$$\overline{R}_{ASP}(N=1)=\overline{R}_{M/M/P}, \quad (\lambda,\cdot,\exp(1/\overline{D}),r=0,\gamma^l).$$

Under nonexponential demands the extension is that $\overline{R}_{ASP}(N=1)=\overline{R}_{M/G/P}$.

Since to the authors' knowledge there does not exist an exact solution for $\overline{R}_{ASP}(N=P)$, an interpolation approximation for $\overline{R}_{ASP}(N=P)$ is developed next by observing the behavior of ASP at extreme ends of system utilization. When $\rho=0$, $\overline{R}_{ASP}(N=P)$ is simply $\overline{S}=\overline{D}/P$ (since execution rates are linear). On the other hand when $\rho\rightarrow 1$ the queue length increases and a waiting job is allocated just one processor upon service (assuming that there are at least as many jobs as free processors). Therefore, for exponential job demands, as $\rho\rightarrow 1$ the system under ASP tends to behave like an M/M/P queue, i.e., $\overline{R}_{ASP}\rightarrow\overline{R}_{M/M/P}$ as $\rho\rightarrow 1$. Combining these two estimates at extreme ends of ρ, we get the following approximation for $\overline{R}_{ASP}(N=P)$ when job demand is exponential. [7]

$$\overline{R}_{ASP}(N=P)\approx(1-\alpha(\rho))\frac{\overline{D}}{P}+\alpha(\rho)\overline{R}_{M/M/P}, \quad \text{under } (\lambda,\cdot,\exp(1/\overline{D}),r=0,\gamma^l,E(j)=\gamma(j)) \quad (12)$$

where $\alpha(0)=0$, $\alpha(1)=1$, and $0<\alpha(\rho)<1$, for $0<\rho<1$.

The following form for $\alpha(\rho)$ is empirically derived by means of curve fitting techniques using simulation estimates of mean system response time for the above workloads at P=10, 20, 50, and 100:

$$\alpha(\rho)=\frac{2}{P}\rho+(0.5-2/P)\rho^s+0.5\rho^{\lceil P/3\rceil},$$

where

$$s=\begin{cases}3.5 & P\leq 20,\\ 4.5 & P=50,\\ 6.0 & P=100.\end{cases}$$

[7] Note that the interpolation approximation on ρ might also be applied for general D, N, and/or γ; however the function $\alpha(\rho)$ is difficult to derive in these cases.

Since approximation (11) will be shown to validate well in Section 4, S_n is the key parameter for job parallelism under the given workload. That is, $\overline{R}_{ASP}$ is approximately the same for all distributions of N that yield the same value for S_n.

The interpolation approximation approach in (11) and (12) also resulted in an accurate approximation when job demand is deterministic ($C_D = 0$) and N has a general distribution, $\mathcal{F}_N$. In this case the functional form of $\alpha(\rho)$ was less carefully constructed and the approximation also has so far been less extensively validated than the approximation for exponential demands ($C_D = 1$). (For the 50 data points validated for this approximation, 42 are with 15% of the simulation estimates and the maximum error is about 28%.) The following summarizes the approximation for $C_D = 0$:

$$\begin{aligned}\overline{R}_{ASP}(\mathcal{F}_N) \approx & \left(\frac{S_n - 1/P}{1 - 1/P}\right)\overline{R}_{ASP}(N=1) + \\ & \left(\frac{1 - S_n}{1 - 1/P}\right)\overline{R}_{ASP}(N=P), \quad \text{under } (\lambda, \cdot, D = \overline{D}, r = 0, \gamma^l, E(j) = \gamma(j)). \end{aligned} \tag{13}$$

where

$$\overline{R}_{ASP}(N=1) = \overline{R}_{M/D/P} \approx \overline{D} + \frac{\rho\sqrt{2(P+1)}}{2(1-\rho)\lambda},$$

and

$$\overline{R}_{ASP}(N=P) \approx (1 - \alpha(\rho))\frac{\overline{D}}{P} + \alpha(\rho)\overline{R}_{M/D/P},$$

$$\alpha(\rho) = \frac{1}{P}\rho + (0.5 - 1/P)\,\rho^{\sqrt{P/2}} + 0.5\,\rho^{P/2}.$$

3.4 PSAPF: $r = 0$

This section first reviews an interpolation approximation from [18] for $\overline{R}_{PSAPF}$ under $(\lambda, \mathcal{F}_N, \mathcal{F}_D, r = 0, \gamma^l, E(j) = \gamma(j))$, then derives a more accurate approximation under the same workload assumptions, and finally provides solutions for constant N and general γ. Section 3.5 derives estimates for $\overline{R}_{PSAPF}$ for $r > 0$.

3.4.1 Previous Analysis for General N: $r = 0$, γ^l

The following interpolation approximation on the pmf of N is shown to provide reasonably accurate estimates of $\overline{R}_{PSAPF}$ under $(\lambda, \mathcal{F}_N, \mathcal{F}_D, r = 0, \gamma^l, E(j) = \gamma(j))$ and is noted to be the same as the interpolation

approximation (7) for $\overline{R}_{FCFS}$ [18].

$$\overline{R}_{PSAPF}(\mathcal{F}_N, r=0) \approx \sum_{k=1}^{P} p_k \overline{R}_{PSAPF}(N=k, r=0), \quad \text{under } (\lambda, \cdot, \mathcal{F}_D^u, \cdot, \gamma^l, E(j)=\gamma(j))$$

$$= \overline{S} + \frac{E\left[\rho^{\sqrt{2(\frac{P}{N}+1)}}\right]}{1-\rho}\left(\frac{1+C_D^2}{2\lambda}\right). \tag{14}$$

In many validations this approximation results in less than 35% errors from simulation estimates. However, it does not validate well in some cases with high C_D and low C_N (e.g., more than 100% relative errors have been observed), which motivates a more accurate approximation for $\overline{R}_{PSAPF}$. Note that approximation (14) gives the "coarse" result that $\overline{R}_{PSAPF}(r=0) \approx \overline{R}_{FCFS}(r=0)$ under the given workload assumptions. The more accurate approximation derived next will enable a more refined comparison.

3.4.2 More Accurate Analysis for General N: $r = 0$, γ^l

A more accurate approximation for $\overline{R}_{PSAPF}$ is derived by observing that PSAPF is essentially a Preemptive Resume (PR) priority scheduling policy. A known heuristic for obtaining performance estimates of PR for a multiserver system with sequential jobs is to compare PR with FCFS in a uniprocessor system and then map the comparison to the multiserver system (cf. [5, 3, 36]). For example, in [3], Buzen and Bondi approximated the mean extra time (i.e., mean response time minus mean service time) of an M/G/c PR queue by

$$\overline{X}_{M/G/c\ PR} \approx \frac{\overline{X}_{M/G/1_c\ PR}}{\overline{X}_{M/G/1_c\ FCFS}} \overline{X}_{M/G/c\ FCFS}, \tag{15}$$

where the $M/G/1_c$ PR queue is obtained by replacing all c servers of the $M/G/c$ PR queue by a single server of power c. (Likewise for FCFS.) A similar heuristic can be used to estimate the mean extra time of a parallel system under PSAPF, $\overline{X}_{PSAPF} \equiv \overline{R}_{PSAPF} - \overline{S}$, as follows.

$$\overline{X}_{PSAPF} \approx \frac{\overline{X}_{M/G/1_P\ PR}}{\overline{X}_{M/G/1_P\ FCFS}} \overline{X}_{FCFS}, \tag{16}$$

where job priorities in the $M/G/1_P$ PR queue are the same as those in the PSAPF system (i.e., inversely proportional to available parallelism).

A closed form expression for $\overline{R}_{PSAPF}$ is derived by obtaining closed form expressions for each of $\overline{X}_{M/G/1_P\ FCFS}$, $\overline{X}_{FCFS}$, and $\overline{X}_{M/G/1_P\ PR}$ in (16). $\overline{X}_{M/G/1_P\ FCFS}$ is simply $\rho^2(1+C_D^2)/(2\lambda(1-\rho))$ [10], and approxima-

tion (7) yields a closed form expression for $\overline{X}_{FCFS} \equiv \overline{R}_{FCFS} - \overline{S}$. The analysis in [11] for an M/G/1 PR queue (under the given workload assumptions) yields,

$$\overline{X}_{M/G/1_P\ PR} = \sum_{k=1}^{P} p_k \left[\frac{\sigma_{k-1}}{1-\sigma_{k-1}} + \frac{\sigma_k}{(1-\sigma_{k-1})(1-\sigma_k)} \left(\frac{1+C_D^2}{2} \right) \right] \frac{\overline{D}}{P}, \quad \text{where} \quad \sigma_k = \rho \sum_{i=1}^{k} p_i. \qquad (17)$$

Thus, under the assumptions $(\lambda, \mathcal{F}_N, \mathcal{F}_D^u, r=0, \gamma^l, E(j)=\gamma(j))$, we have the following closed form expression for $\overline{R}_{PSAPF} = \overline{S} + \overline{X}_{PSAPF}$:

$$\overline{R}_{PSAPF}(r=0) \approx \overline{S} + \left\{ \sum_{k=1}^{P} p_k \left[\frac{\sigma_{k-1}}{1-\sigma_{k-1}} + \frac{\sigma_k}{(1-\sigma_{k-1})(1-\sigma_k)} \left(\frac{1+C_D^2}{2} \right) \right] \frac{\overline{D}}{P} \right\} E\left[\rho^{\sqrt{2(\frac{P}{N}+1)}-2} \right]. \qquad (18)$$

Note that the accuracy of approximation (18) can be improved by using a more accurate approximation for $\overline{X}_{FCFS}$ in (16); however, the use of numerical analysis entails significant loss of insight [18].

3.4.3 Analysis under Constant N

When available parallelism is constant, i.e., $N = k$, PSAPF is identical to FCFS and approximation (8) is valid for the system $(PSAPF, \lambda, N=k, \mathcal{F}_D, r=0, \gamma)$ as well. Thus,

$$\overline{R}_{PSAPF}(N=k, r=0) \approx \frac{\overline{D}}{\gamma(k)} + \frac{\nu\sqrt{2(P/k+1)}}{1-\nu} \left(\frac{1+C_D^2}{2\lambda} \right), \quad \text{under } (\lambda, \cdot, \mathcal{F}_D, \cdot, \gamma, E(j)) \qquad (19)$$

where $\nu = \dfrac{\lambda \overline{D}}{P} \cdot \dfrac{k}{\gamma(k)}$.

Note that as in the case of FCFS, approximations (18) and (19) for $\overline{R}_{PSAPF}$ increase linearly in C_D^2.

3.5 PSAPF: $r > 0$

The estimate $\overline{R}_{PSAPF}$ is first derived for fully correlated workloads ($r = 1$) and then combined with approximation (18) for uncorrelated workloads ($r = 0$) to yield an estimate for arbitrary partial correlation ($0 < r < 1$).

3.5.1 Analysis for $r = 1$: γ^l

The approximation for $\overline{R}_{PSAPF}$ under (λ, $\mathcal{F}_N$, $\mathcal{F}_D^u$, r=1, γ^l, $E(j) = \gamma(j)$) is derived by: (1) classifying jobs according to their available parallelism, (2) computing the mean response time for each class of jobs

by approximating the *average interference* from other classes of jobs, and (3) computing the overall mean response time as a weighted sum of the approximate mean response times per class. This general approach yields very accurate estimates of $\overline{R}_{EQS}$ under the given workload conditions [19]. In the case of PSAPF the particular approximate representation of average interference by other job classes yields a system for each class that reduces to a preemptive resume queue, from which the class mean response time is computed.[8]

Let a job with available parallelism k belong to class C_k, for $k = 1, \ldots, P$. Let $\overline{R}_{PSAPF,C_k}$ denote the mean response time of class C_k in the system $(PSAPF, \lambda, \mathcal{F}_N, \mathcal{F}_D^u, r, \gamma, E(j) = \gamma(j))$. Clearly,

$$\overline{R}_{PSAPF} = \sum_{k=1}^{P} p_k \overline{R}_{PSAPF,C_k}. \tag{20}$$

The approximate processor contention from classes other than C_k is modeled by assuming each such class has available parallelism k, but retains its total service requirements and job priority as before. More precisely, we approximate $\overline{R}_{PSAPF,C_k}$ to be the mean response time of class C_k in a system Γ_k which is like the original system except that a class C_j job in Γ_k has demand D_j, priority j, and available parallelism k, where $\overline{D}_j = \dfrac{\overline{D}}{\overline{N}} \cdot j$, as per the correlation model in Section 2.2. The instantaneous load of class C_j jobs is not accurately modeled by assuming that class C_j jobs have parallelism k. However, the priority and offered load of class C_j jobs are accurately modeled. Thus, the overall interference of C_j with C_k may be reasonably well represented.

An approximation for $\overline{R}_{PSAPF,C_k}$ is derived by solving for the mean response time of class k in system Γ_k. Since jobs from classes $k+1$ to P have lower priority than k it is only necessary to consider arrivals from classes 1 through k to obtain $\overline{R}_{\Gamma_k,C_k}$. Recall that in Γ_k all jobs have an available parallelism of k. First assume that k evenly divides P. Thus processors are allocated or preempted in units of k at a time. If processors are grouped k at a time and each such cluster is thought of as a superprocessor, then we realize that Γ_k essentially functions as an $M/G/c$ PR queue with $c = P/k$ servers each of power k, and with k priority classes. Therefore, $\overline{R}_{\Gamma_k,C_k}$ is equal to the mean response time of the k^{th} priority class in this $M/G/c\ PR$ queue. Tabetaeoul and Kouvatsos [36] derive an approximation for per class mean response times of a $GI/G/c\ PR$ queue using a heuristic similar to (15). Using their heuristic we obtain the following

[8] This general approach validates well not only for $r = 1$ but also for $0 \leq r < 1$. However, the separate approximations for $\overline{R}_{PSAPF}(r = 0)$ and $\overline{R}_{PSAPF}(r)$, $0 < r < 1$, yield more insight.

expression for $\overline{R}_{\Gamma_k,C_k}$, which is derived in Appendix A.

$$\overline{R}_{\Gamma_k,C_k} \approx c\,\overline{x}_k + \frac{1}{p_k}\left(\sum_{i=1}^{k-1} g_i\right)\left(\sigma_k^{\sqrt{2(c+1)}-2} - \sigma_{k-1}^{\sqrt{2(c+1)}-2}\right) + \frac{1}{p_k} g_k \sigma_k^{\sqrt{2(c+1)}-2}, \quad p_k > 0, \qquad (21)$$

where

$$g_i = p_i \frac{\sigma_{i-1}}{1-\sigma_{i-1}}\overline{x}_i + \frac{\lambda p_i \sum_{j=1}^{i}\{p_j \overline{x}_j^2\}}{(1-\sigma_{i-1})(1-\sigma_i)}\left(\frac{1+C_v^2}{2}\right),$$

$$\overline{x}_i = \frac{\overline{D}i}{\overline{N}\overline{P}}, \quad \text{and} \quad \sigma_i = \lambda \sum_{j=1}^{i} p_i \overline{x}_i, \quad i = 1, \ldots, k.$$

Approximation (21) can be applied even when k does not evenly divide P to obtain

$$\overline{R}_{PSAPF} \approx \sum_{k=1}^{P} p_k\, h(k, \lambda, (p_1, \ldots, p_k), (\overline{D}, C_v)), \quad \text{under } (\lambda, \mathcal{F}_N, \mathcal{F}_D^u, r = 1, \gamma^l, E(j) = \gamma(j)), \qquad (22)$$

where $h(k, \lambda, (p_1, \ldots, p_k), (\overline{D}, C_v))$ is given by the RHS of (21). Note from (21) that $\overline{R}_{PSAPF}$ grows linearly in the squared coefficient of variation of demand, C_v, of each job class when $r = 1$ and $\gamma = \gamma^l$.

3.5.2 Analysis for $0 < r < 1$: γ^l

Thus far estimates have been derived for $\overline{R}_{PSAPF}(r = 0)$ and $\overline{R}_{PSAPF}(r = 1)$. To estimate $\overline{R}_{PSAPF}$ for a general r between 0 and 1, consider an interpolation approximation on r. That is,

$$\overline{R}_{PSAPF}(r) \;\approx\; (1 - f(r))\,\overline{R}_{PSAPF}(r = 0) \;+\; f(r)\,\overline{R}_{PSAPF}(r = 1),$$

where $f(r)$ is a suitable function of r. Note from (2) that as $\rho \to 0$, $\overline{R}_{PSAPF} = \overline{S} = (1 - r^2)\overline{S}(r = 0) + r^2\overline{S}(r = 1)$. Therefore as $\rho \to 0$, $f(r) = r^2$. Validations show that for $\rho > 0$ this choice of $f(r)$ continues to yield accurate estimates of $\overline{R}_{PSAPF}$. Therefore,

$$\overline{R}_{PSAPF}(r) \approx (1 - r^2)\overline{R}_{PSAPF}(r = 0) \;+\; r^2\overline{R}_{PSAPF}(r = 1), \quad \text{under } (\lambda, \mathcal{F}_N, \mathcal{F}_D^u, \cdot, \gamma^l, E(j) = \gamma(j)), \qquad (23)$$

where $\overline{R}_{PSAPF}(r = 0)$ and $\overline{R}_{PSAPF}(r = 1)$ are estimated using approximations (18) and (22), respectively.

Note that the form of approximation (23) is identical to (10), which was proposed for the EQS policy. This will prove useful in comparing the performance of the EQS and PSAPF policies in the range $0 < r < 1$.

4 Validations of Approximations for $\overline{R}_{ASP}$ and $\overline{R}_{PSAPF}$

This section presents results of validation experiments for the approximations for $\overline{R}_{ASP}$ and $\overline{R}_{PSAPF}$ derived in Section 3. The parameter values for the validations are as follows:

- For most of validations P=20 or P=100 processors.[9]
- Three different distributions for available parallelism N were used. First, the bounded-geometric distribution defined in Section 2.5. Second, a uniform distribution with several values for the lower and upper limits. Third, constant N, i.e., $N = k$.
- The ASP approximations (12) and (11) were validated using an exponential distribution for demand, and the PSAPF approximations were validated using exponential ($C_v = 1$) demands as well as two-stage hyperexponential (H_2) demands with $C_v = 5$. In a few cases the PSAPF approximations were also validated for deterministic and Gamma distributions of demand. The accuracy of the approximations for deterministic demands was nearly the same as the accuracy for exponential demands and for the Gamma distribution the accuracy was the same as for H_2 demands with the same C_v. A few test cases for $C_v < 5$ also showed that the accuracy of the PSAPF approximations generally improves when C_v is decreased.
- For all validations $\overline{D}$ was set to P so that $\rho \equiv \lambda\overline{D}/P = \lambda$. In the validations ρ was varied from 0.1 to 0.9.

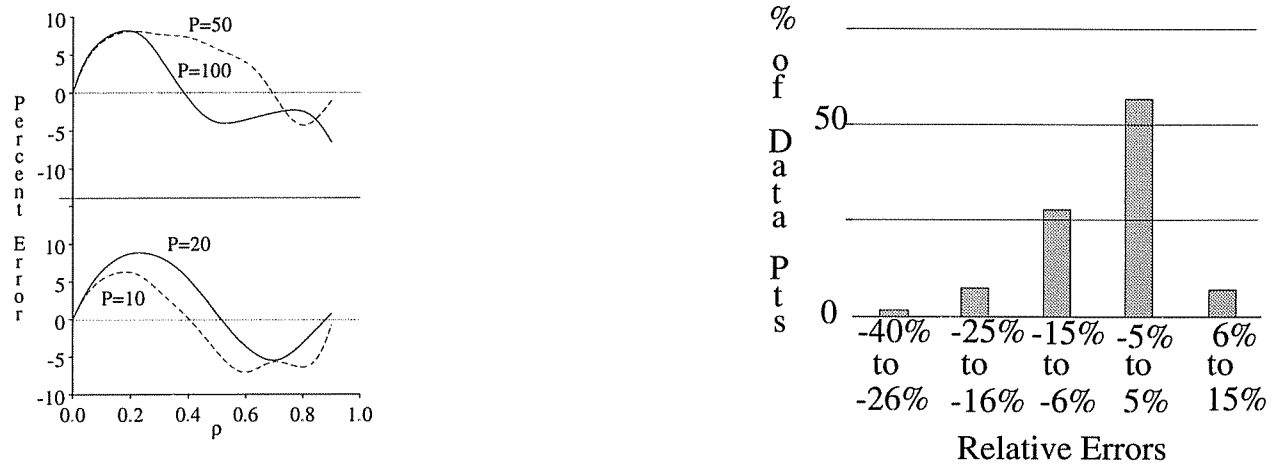
The approximations for $\overline{R}_{PSAPF}$ for constant available parallelism were validated using exact matrix-geometric analysis [27, 24]. In all other cases, the approximations were validated using discrete event simulation. All simulation estimates of mean response time had 95% confidence intervals with less than 10% half-widths, and in nearly all cases the half-widths were less than 5%. The batch means method was used if obtaining the regenerative cycles was too time consuming.

4.1 ASP Validations

Figure 3a depicts the relative errors for approximation (12) for systems with 10, 20, 50, and 100 processors. Observe that for all four system sizes approximation (12) overestimates $\overline{R}_{ASP}(N = P)$ at low utilizations, but underestimates $\overline{R}_{ASP}(N = P)$ at moderate to high utilizations. However, in all cases the relative errors are less than 10%. Approximation (11) can be expected to have higher relative errors since it uses approximation (12). To validate approximation (11) simulation experiments were run for many bounded-geometric distributions of N with different values of P_{max} and p, and for uniform and constant distributions of N. The total number of data points in the validations (excluding the points for $N = P$) was about 140 for systems with 20 and 100 processors. Figure 3b summarizes these validation results by plotting histograms

[9] In some cases systems with 10 or 50 processors and in some other cases systems with 500 or 1000 processors were considered. The accuracy of the approximations was approximately the same in these cases as the accuracy for 20 or 100 processors.

of relative error. The figure shows that approximation (11) is very accurate. For more than 90% of the data points the approximation is within 15% of the simulation estimates. The largest error (-36.4%) occurred for a U[50,100] distribution for N at $\rho = 0.3$. In general, the errors are larger and more negative for workloads with high average available parallelism (say $\overline{N} > 3P/4$) when load is low to moderate ($\rho \leq 0.5$).



(a) Approximation for N=P

(b) Approximation for general N

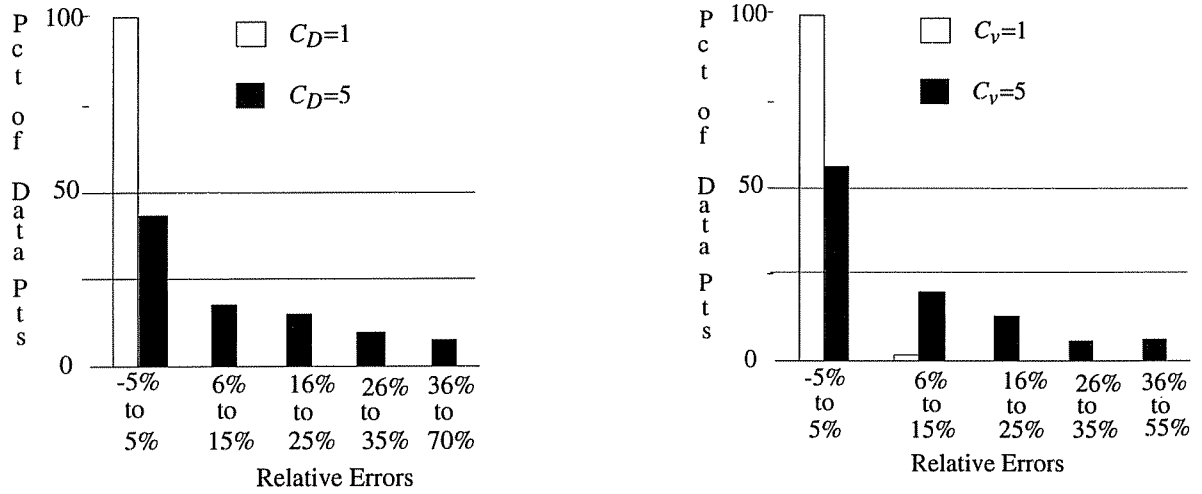
Figure 3: Validations of ASP Approximations

4.2 PSAPF Validations

Validations of approximations (18), (22), and (23) will be presented. For approximation (18), and $r = 0$, a total of 612 data points were validated, 306 for each of $C_v = 1$ and $C_v = 5$. Figure 4a presents the histograms of relative errors for this approximation. The results indicate that the approximation is extremely accurate when $C_v = 1$, and reasonably accurate at $C_v = 5$ (about 90% of the data points have less than 35% error at $C_v = 5$). The approximation is more conservative (i.e., only very small negative relative errors have been observed) and more accurate than the PSAPF approximation in [18]. The maximum relative error occurred at $C_v = 5$, $N = 3/4P$, and $\rho = 0.2$ for both P=20 and P=100. In general, the largest errors at $C_v = 5$ were observed for distributions of N with moderate to high $\overline{N}$ and low C_N. Recall also that the overall accuracy might be improved if a more accurate approximation for $\overline{X}_{FCFS}$ is used in approximation (18).

Approximation (22) is validated against simulation estimates for 356 data points, 178 each for $C_v = 1$ and $C_v = 5$. (Since the constant N distribution need not be validated when $r = 1$ the total number of validations is fewer than when $r = 0$.) Figure 4b summarizes the validations for this approximation. As seen

from the figure approximation (22) is very accurate at low C_v and reasonably accurate at high C_v (about 95% of the data points have less than 35% error when $C_v = 5$). The maximum error at $C_v = 5$ occurred for the data point N=U[1,100], $\rho = 0.5$. The approximation errors when $r = 1$ were highest for low C_N workloads at low to moderate load.



(a) Approximation for r=0

(b) Approximation for r=1

Figure 4: Relative Error Histograms for PSAPF Approximations: $r = 0$ and $r = 1$

For approximation (23) the validations consist of a total of 452 data points, 226 data points for each of $C_v = 1$ and $C_v = 5$ (excluding the cases for $r = 0$ and $r = 1$). Three values of r were included in the experiments, viz., $r = 0.25$, $r = 0.5$, and $r = 0.75$. Figure 5 displays histograms of the relative error at $C_v = 1$ and $C_v = 5$. Again, the accuracy of approximation (23) is very high at $C_v = 1$ and reasonable at $C_v = 5$. Thus all three approximations for PSAPF (i.e., for r=0, r=1, and for general r) are reasonably accurate in general, as long as $C_v \leq 5$. The maximum observed error for approximation (23) was for a specific bounded-geometric distribution for N with low C_N, $C_v = 5$, $r = 0.5$, and $\rho = 0.8$. Thus, as for the approximations at $r = 0$ and $r = 1$, approximation (23) is more accurate for distributions of N with high C_N.

5 Policy Comparison Results

The goal of this section is to compare the performance of ASP, EQS, FCFS, and PSAPF under the general workload assumptions $(\lambda, \mathcal{F}_N, \mathcal{F}_D^u, r, \gamma, E(j) = \gamma(j))$. The mean response times of ASP, FCFS, and PSAPF

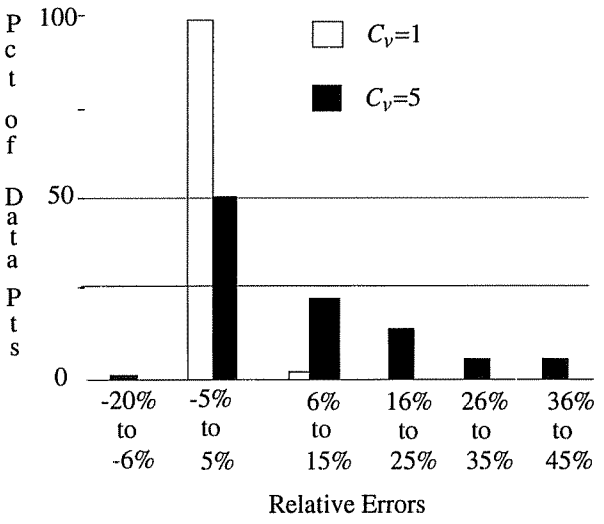


Figure 5: Relative Error Histograms for PSAPF Interpolation on r

$r = 0.25,\ 0.5,\ 0.75$

were each derived under one or more restrictive assumptions, i.e., linear execution rates for all three policies, no correlation for ASP and FCFS, and exponential demands for ASP. However, if it turns out that the restrictive assumptions are more favorable to one policy, Ψ_1, over another, Ψ_2, and yet Ψ_1 performs worse, then the same relative ordering between Ψ_1 and Ψ_2 will hold under more general conditions that are less favorable to Ψ_1. In this way we will be able to generalize the results from comparisons of the four policies under the restrictive assumptions.

The following theorem will prove useful in understanding the impact of execution rate assumptions on policy comparisons. This theorem shows that for any fixed set of jobs with a common workload ERD, γ, the total execution rate of all jobs (or equivalently the processor efficiency) is maximum for the EQS policy.

Theorem 5.1 *Consider a set of K jobs with available parallelisms $(n_1, \ldots, n_K)$. Let Ψ be a processor allocation policy that allocates a_i^{Ψ} processors to job i, for $i = 1, \ldots, K$. Then for a workload ERD γ that is concave and nondecreasing, and for $E(j) = \gamma(j)$, i.e., jobs dynamically and efficiently redistribute their work,*

$$\sum_{i=1}^{K} E(a_i^{EQS}) \geq \sum_{i=1}^{K} E(a_i^{\Psi}), \quad \textit{for any processor allocation policy } \Psi. \tag{24}$$

Proof. See Appendix B. ∎

Remark: An extension to Theorem 5.1 is that the available parallelisms can be random variables $(N_1, \ldots, N_K)$,

in which case one should take the expected value of the sums in (24).

The intuition behind the result is that when γ is concave the total execution rate decreases with variability in allocation. EQS tends to allocate an equal fraction of processors to jobs and this leads to high overall efficiency. As per the theorem, the assumption of the linear ERF is more favorable to the ASP, FCFS, and PSAPF policies as compared to EQS. We discuss the favorability of other workload parameter settings as they arise in the comparisons below.

This section first compares ASP and EQS and shows that EQS performs as well or better than ASP for essentially the entire parameter space. We then compare FCFS and PSAPF and show that PSAPF outperforms FCFS for most of the parameter space. Section 5.3 compares EQS and PSAPF and delineates the regions under which each policy performs best. Section 5.4 shows how the performance comparison results in this paper generalize and unify previous work. Note that for all experiments in this section $\overline{D} = P$ and thus $\rho \equiv \lambda\overline{D}/P = \lambda$.

5.1 ASP versus EQS

Section 5.1.1 compares the performance of ASP and EQS for uncorrelated workloads with linear execution rates. The comparison is made using approximation (9) for $\overline{R}_{EQS}$. For $\overline{R}_{ASP}$, approximation (11) is used for exponential job demands, approximation (13) is used for $C_D = 0$, and simulation is used for $C_D > 1$. In all cases, the linear ERF is most favorable to the ASP policy, which allows extrapolation of the policy comparisons to sublinear ERFs. Section 5.1.2 uses simulation for $\overline{R}_{ASP}$ to compare the performance of ASP and EQS.

5.1.1 ASP versus EQS: r=0

Key to the comparison of $\overline{R}_{ASP}$ and $\overline{R}_{EQS}$ for uncorrelated workloads with exponential job demands and linear execution rates, i.e., $(\lambda, \mathcal{F}_N, \exp(1/\overline{D}), r = 0, \gamma^l, E(j) = \gamma(j))$, are the following observations from (9) and (11). First, for a given system size, S_n, $\overline{D}$, and ρ are the key determinants of $\overline{R}_{ASP}$ and $\overline{R}_{EQS}$ under the given assumptions. Second, for fixed S_n and ρ the ratio $\overline{R}_{ASP}/\overline{R}_{EQS}$ is insensitive to $\overline{D}$ because each of the formulas in (9) and (11) is directly proportional to $\overline{D}$. Therefore, if $\overline{D}$ is held fixed and the ratio $\overline{R}_{ASP}/\overline{R}_{EQS}$ is plotted as a function of S_n for different values of ρ, the results will hold for all $\overline{D}$ and all distributions of N in the assumed workloads.

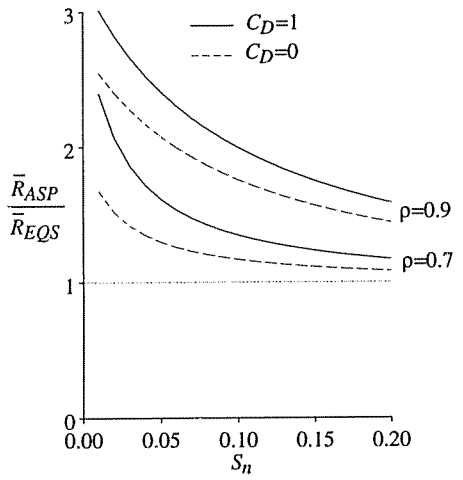
Consider the maximum value of S_n, i.e, $S_n = 1$, in which case all jobs are fully sequential. When $N = 1$, the EQS system is identical to an M/M/P processor sharing (PS) system and thus $\overline{R}_{EQS}(N = 1) = \overline{R}_{M/M/P\ PS}$. On the other hand ASP is identical to FCFS when $N = 1$ and thus for exponential demands $\overline{R}_{ASP}(N = 1) = \overline{R}_{M/M/P\ FCFS}$. Since $\overline{R}_{M/M/P\ PS} = \overline{R}_{M/M/P\ FCFS}$ [31], $\overline{R}_{ASP} = \overline{R}_{EQS}$ for $C_D = 1$, $r = 0$, and $S_n = 1$.[10]

Next consider how these policies compare as job parallelism increases, that is, as S_n decreases. Figure 6a plots $\overline{R}_{ASP}/\overline{R}_{EQS}$ versus S_n for the workload $(\lambda, \mathcal{F}_N, \exp(1/\overline{D}), r = 0, \gamma^l, E(j) = \gamma(j))$. Consider only the solid curves in the figure, for $C_D = 1$, for now. The range of S_n in Figure 6a covers the likely practical values of $\overline{N}$ (i.e., $\overline{N} = 0.05P$ to P). Recall that when $S_n = 1$ (not shown) $\overline{R}_{EQS} = \overline{R}_{ASP}$ and thus the ratios for $C_D = 1$ will converge to 1 when $S_n = 1$. Furthermore, since the linear ERF assumption results in the lowest possible ratio of $\overline{R}_{ASP}/\overline{R}_{EQS}$ at each value of ρ, as per Theorem 5.1, the ratios for workloads with sublinear ERFs will lie above the ratios shown for the linear ERF. Figure 6a reveals that over the entire range of $S_n < 1$, the EQS policy outperforms the ASP policy. The ASP policy becomes more competitive with the EQS policy as S_n increases, but is significantly less competitive for workloads that are (nearly) fully parallel. The reason for the poor performance of ASP is its lack of flexibility in processor allocation. Unlike the dynamic allocation under EQS, the (adaptive) static allocation under ASP can leave processors idle when parallel jobs could otherwise use them.

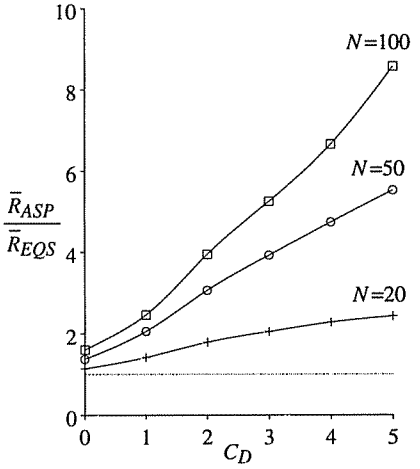
To compare the policies for distributions of demand other than the exponential, first note from (9) that for fixed $\overline{D}$, $\overline{R}_{EQS}$ is insensitive to $\mathcal{F}_D$. On the other hand, at $S_n = 1$, i.e., N=1, and $\gamma = \gamma^l$, $\overline{R}_{ASP} = \overline{R}_{M/G/P}$ and thus ASP policy performance is sensitive to C_D. When P is large (say $P \geq 100$), and $S_n = 1$, $\overline{R}_{ASP}$ is slightly smaller than $\overline{R}_{EQS}$ for $C_D < 1$, equal to $\overline{R}_{EQS}$ at $C_D = 1$, and then increases with respect to $\overline{R}_{EQS}$ with further increase in C_D. The intuition for the increase of $\overline{R}_{ASP}$ with respect to C_D at $S_n = 1$ is that a scheduled job runs to completion without interruption and each large demand job in execution reduces the number of system processors available for serving small jobs. This intuition should also apply for $S_n < 1$.

Figure 6a also contains the results for $\overline{R}_{ASP}/\overline{R}_{EQS}$ versus S_n when $C_D = 0$ (using approximations (13) and (9)). As suggested by intuition, the ratios are lower than when $C_D = 1$. However, the ratio is greater than one throughout the range of S_n shown in the figure and will become only marginally smaller than 1 at $S_n = 1$, as noted above.

[10]Note that $\overline{R}_{EQS}(C_D = 1) = \overline{R}_{EQS}(\exp(1/\overline{D}))$, since the mean response time for EQS depends only on the first moment of the job demand distribution. We further surmise, based on simulation experiments, that $\overline{R}_{ASP}(C_D = 1) \approx \overline{R}_{ASP}(\exp(1/\overline{D}))$.



(a) $\overline{R}_{ASP}/\overline{R}_{EQS}$ versus S_n, $C_D = 0,\ 1$



(b) $\overline{R}_{ASP}/\overline{R}_{EQS}$ versus C_D, $\overline{D} = P$

Figure 6: $\overline{R}_{ASP}/\overline{R}_{EQS}$: $r = 0$, γ^l

$P = 100$

For $C_D > 1$ we do not have analytic estimates of $\overline{R}_{ASP}$ and thus we use simulation to show the trends in relative policy performance.[11] As before, approximation (9) for $\overline{R}_{EQS}$ is valid for all C_D. Figure 6b plots $\overline{R}_{ASP}/\overline{R}_{EQS}$ versus C_D for constant available parallelism and two-stage hyperexponential (H_2) demand distributions. The figure shows that $\overline{R}_{ASP}$ increases significantly with C_D and the rise is sharper for larger available parallelism. The intuition for the latter observation is that jobs with higher parallelism and larger processing demand can occupy a larger number of servers, thus more significantly reducing the processors available to serve waiting jobs. Using the same intuition it appears likely that $\overline{R}_{ASP}$ should increase with C_D for general demand and parallelism distributions. This was partially verified for specific nondeterministic distributions of N (not shown).

Thus, for uncorrelated workloads it appears that $\overline{R}_{EQS} \leq \overline{R}_{ASP}$ except for S_n close to 1, $C_D = 0$, and linear execution rates, and even for these extreme parameter values, $\overline{R}_{ASP}$ is only marginally smaller than $\overline{R}_{EQS}$ for systems with a moderate to large number of processors.

Before concluding this section, it may be of interest to compare how EQS performs for workloads with sublinear γ versus how ASP performs when γ is linear. Assuming that job demand is exponential, $S_n(\gamma)$

[11] All simulation experiments in this paper have 95% confidence intervals with less than 10% half-widths, and in almost all cases the half-widths are less than 5% of the estimate. The confidence intervals were generated using the regenerative method whenever feasible and otherwise the method of batch means.

and $S_n(\gamma^l)$ are the respective key parallelism parameters for EQS and ASP. For the ERD in (5) $S_n(\gamma^l)$ is related to $S_n(\gamma)$ by (6), and thus $S_n(\gamma)$ uniquely determines the performance of both policies. For $\rho = 0.7$ and $\rho = 0.9$ Figure 7 plots the ratio $\overline{R}_{ASP}(\gamma^l)/\overline{R}_{EQS}(\gamma)$ versus $S_n(\gamma)$ for the ERD (5) with $\beta = 100$, which is considerably sublinear as shown in Figure 2. The ratios converge to 1 when $S_n = 1$ and are thus greater than 1 throughout the range of $S_n < 1$. For (low) values of ρ where mean service time dominates mean response time, the ratio will be less than 1 (except at $S_n = 1$). However, Figure 7 shows that (at moderate to high loads) a poor choice of scheduling policy, perhaps dictated by existing system software or hardware can be more detrimental to overall mean system response time than parallel program overheads.

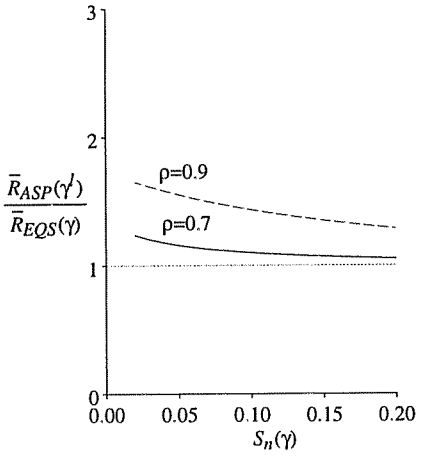


Figure 7: $\overline{R}_{ASP}(\gamma^l)/\overline{R}_{EQ}(\gamma)$ versus $S_n(\gamma)$: r=0

P=100, $C_D = 1$, $\beta = 100$

5.1.2 ASP versus EQS: r=1

In Section 5.1.1 it was noted that lack of flexibility of processor allocation under ASP causes it to perform worse than EQS when $r = 0$. For example, if a highly parallel job is allocated fewer processors than its available parallelism it cannot make use of additional processors when they become idle. Conversely, if a fully parallel job that has large processing demand is allocated all of the processors, it may block lower demand jobs for a significant length of time. When $r = 1$, the more parallel jobs also have larger demands and therefore one might expect that the static allocation under ASP will be even more detrimental – i.e.,

the differential between $\overline{R}_{ASP}$ and $\overline{R}_{EQS}$ will increase with r.

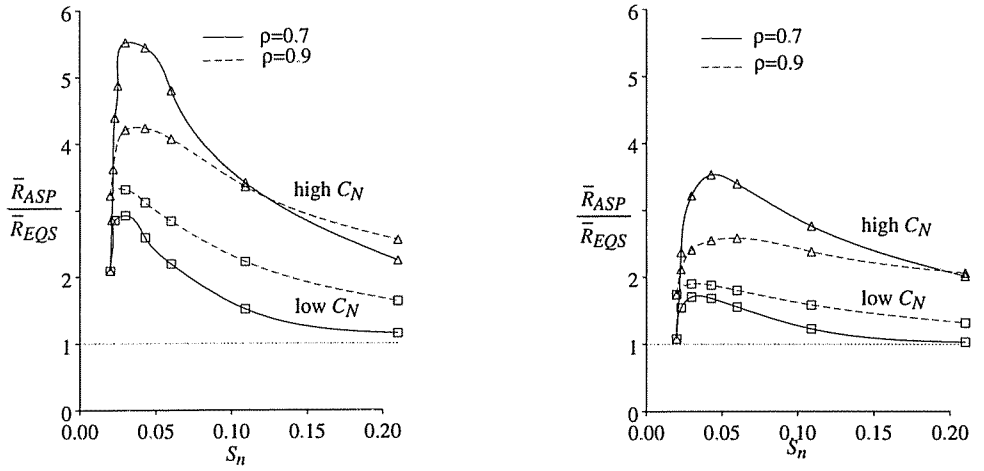
$\overline{R}_{ASP}$ and $\overline{R}_{EQS}$ will be compared under the assumptions $(\lambda, \cdot, \mathcal{F}_D^u = \exp(1/\overline{D}), r = 1, \gamma^l)$, using simulation to estimate $\overline{R}_{ASP}$. The assumption of linear ERFs is more favorable to ASP, as is the assumption of exponential demands compared with $C_v > 1$. Due to lack of analytic estimates it is unknown whether S_n is a key parameter for $\overline{R}_{ASP}$. As a result, the performance of ASP will be compared with that of EQS for the high C_N and low C_N distributions for N defined in Section 2.5, and thus estimate the range of relative policy performance between these two extremes of bounded-geometric distributions.

Figure 8a plots $\overline{R}_{ASP}/\overline{R}_{EQS}$ versus S_n for the high C_N and low C_N workloads for the linear ERF at two values of ρ. Note that the results in fact show that S_n is not by itself the key parallelism determinant of ASP mean response time. The curves for workloads with sublinear ERFs will lie above those shown for the linear ERF. We note from Figure 8a that $\overline{R}_{ASP}/\overline{R}_{EQS}$ is higher for the high C_N workload than the low C_N workload. Moreover, the ratios for the high C_N workload are markedly higher than the ratios in Figure 6. What causes the performance of ASP to degrade at $r = 1$ when C_N is high? The intuition for this behavior is as follows. For all data points in Figure 8, noting that $C_D = 1$, the mean waiting time (not shown) under ASP is negligible compared to the overall mean response time of ASP[12]. Thus for the given range of utilizations and for the given workload assumptions, the mean response time of an arriving job in the ASP system is primarily determined by the number of processors it is allocated when it begins service. The high C_N workload has a much higher percentage of fully parallel jobs as compared to the low C_N workload (see Section 2.5) and fully parallel jobs under ASP have the highest mean service time among all parallelism classes. Furthermore, these jobs are frequently allocated a smaller fraction of the processors than they can productively use. This phenomenon is more detrimental for ASP as compared to EQS since under ASP a job's partition cannot expand beyond its initial allocation.

From Figure 8a we also observe that all curves initially increase sharply with S_n, reach a peak at moderate parallelism, and then decrease with further increase in S_n. This behavior is explained by two opposing trends that occur when S_n increases. The first trend is that when S_n increases the mean demand of highly parallel jobs increases[13] which causes their mean response time under ASP to increase relative to EQS because a highly parallel job can make use of idle processors under EQS but not under ASP. The second trend is that

[12] This observation does not concur with the observations of Setia and Tripathi [33] because we examine a system with $P = 100$ whereas they examined systems with $P \leq 10$ in which it is less likely for a job to find an idle processor upon arrival.

[13] As S_n increases more and more of the probability mass shifts to lower values of parallelism where jobs have smaller mean demands (since $r = 1$). Therefore, to keep the overall mean demand as $\overline{D}$ the mean demand of highly parallel jobs increases.



(a) linear ERF (b) $\beta = 100$

Figure 8: $\overline{R}_{ASP}(\gamma^l)/\overline{R}_{EQ}(\gamma)$ versus $S_n(\gamma)$: r=1

P=100, $\overline{D} = P$, $C_v = 1$

when S_n increases the percentage of fully parallel jobs decreases which decreases their contribution to overall mean response time. When S_n is low the first trend dominates causing the curves in Figure 8 to increase and as S_n increases further the second trend dominates causing the curves to decrease.

As in the case of $r = 0$ we also plot $\overline{R}_{ASP}(\gamma^l)/\overline{R}_{EQS}(\gamma)$ versus $S_n(\gamma)$ for the highly sublinear ERD with $\beta = 100$. Figure 8b shows that the ratios are greater than 1 throughout the range of S_n (they converge to 1 at $S_n = 1$). At lower utilizations where $\overline{S}$ dominates mean response time, the ratios will be less than 1. Thus at moderate to high utilizations EQS performs better even when the EQS workload has a sublinear ERF and the ASP workload has the linear ERF.

To summarize the policy comparison results for ASP and EQS, we conclude that EQS has significantly better performance over essentially the entire system parameter space because (1) it utilizes processors better, that is, jobs make use of idle processors whenever possible, and (2) its mean response time is not sensitive to variation in job demand. ASP becomes more competitive with EQS as S_n decreases, correlation decreases, C_v decreases, and ERF linearity increases. While the last three observations follow from intuition and from Theorem 5.1, the first observation follows from the results shown in Figure 6 (and 8) and would be difficult to obtain in the absence of simple approximations such as (11) and (13) for $\overline{R}_{ASP}$.

5.2 PSAPF versus FCFS

The purpose of this section is to quantify the difference in performance between PSAPF and FCFS over the model parameter space $(\lambda, \mathcal{F}_N, \mathcal{F}_D^u, r, \gamma, E(j) = \gamma(j))$. When N is deterministic, PSAPF is identical to FCFS. For nondeterministic N, however, we expect PSAPF to perform differently than FCFS. In general, one can expect PSAPF to perform better than FCFS for three reasons. First, by delaying service of more parallel jobs, PSAPF tends to keep processor utilization high for a larger portion of each busy period [1]. Second, for correlated workloads PSAPF gives higher priority to jobs with smaller mean demands. Third, at high instantaneous load the overall efficiency is higher under PSAPF for sublinear γ because jobs that receive higher priority also execute more efficiently. Due to the last two reasons the most favorable parameter values for FCFS relative to PSAPF are no correlation ($r = 0$) and the linear ERF ($\gamma = \gamma^l$). Using the analytic models of Section 3 we show how the policies compare under these favorable conditions and extrapolate the results to the case of sublinear ERFs. We then provide simulation data to show how PSAPF and FCFS compare under correlated workloads.

5.2.1 PSAPF versus FCFS: r=0

Consider the workloads with $r = 0$ and $\gamma = \gamma^l$. Using approximation (16) we have that

$$\overline{X}_{PSAPF} \approx \overline{X}_{FCFS} \times \frac{\overline{X}_{M/G/1_P\ PR}}{\overline{X}_{M/G/1_P\ FCFS}}.$$

To compare $\overline{R}_{PSAPF}$ with $\overline{R}_{FCFS}$ we need to compare $\overline{X}_{M/G/1_P\ PR}$ with $\overline{X}_{M/G/1_P\ FCFS}$. From Section 3.4.2, under the given workload

$$\overline{X}_{M/G/1_P\ PR} = \sum_{k=1}^{P} p_k \left[\frac{\sigma_{k-1}}{1-\sigma_{k-1}} + \frac{\sigma_k}{(1-\sigma_{k-1})(1-\sigma_k)} \left(\frac{1+C_D^2}{2} \right) \right] \frac{\overline{D}}{P}, \quad \text{where} \quad \sigma_k = \rho \sum_{i=1}^{k} p_i,$$

and

$$\overline{X}_{M/G/1_P\ FCFS} = \frac{\rho}{1-\rho} \left(\frac{1+C_D^2}{2} \right) \frac{\overline{D}}{P}.$$

Note that relative policy performance is only sensitive to the first two moments of D. When $D = \exp(1/\overline{D})$ then $\overline{X}_{M/G/1_P\ PR} = \overline{X}_{M/M/1_P}$ and thus for all $\mathcal{F}_D$ with $C_D = 1$ and fixed $\overline{D}$, we have $\overline{X}_{M/G/1_P} = \overline{X}_{M/M/1_P}$

which means that $\overline{X}_{PSAPF} \approx \overline{X}_{FCFS}$ when $C_D = 1$. Setting $C_D = 1$ in the formulas for $\overline{X}_{M/G/1_P\ PR}$ and $\overline{X}_{M/G/1_P\ FCFS}$ we find that

$$\sum_{k=1}^{P} p_k \left[\frac{\sigma_{k-1}}{1-\sigma_{k-1}} + \frac{\sigma_k}{(1-\sigma_{k-1})(1-\sigma_k)} \right] \frac{\overline{D}}{P} = \frac{\rho}{1-\rho} \frac{\overline{D}}{P}.$$

Now consider $C_D > 1$. Since $(1 + C_D^2)/2 > 1$, we obtain

$$\overline{X}_{M/G/1_P\ PR} < \left(\frac{1+C_D^2}{2} \right) \sum_{k=1}^{P} p_k \left[\frac{\sigma_{k-1}}{1-\sigma_{k-1}} + \frac{\sigma_k}{(1-\sigma_{k-1})(1-\sigma_k)} \right] \frac{\overline{D}}{P} = \left(\frac{1+C_D^2}{2} \right) \frac{\rho}{1-\rho} \frac{\overline{D}}{P} = \overline{X}_{M/G/1_P}.$$

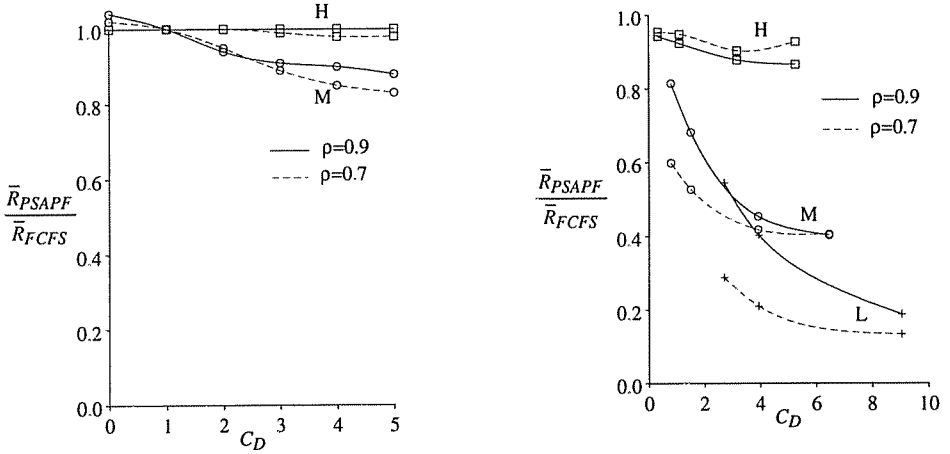
Thus $\overline{X}_{PSAPF} < \overline{X}_{FCFS}$ when $C_D > 1$. Likewise, when $C_D < 1$, $\overline{X}_{PSAPF} > \overline{X}_{FCFS}$. Thus, at $r = 0$, and $\gamma = \gamma^l$, the relative performance of PSAPF and FCFS as determined by C_D is as follows

$$\overline{R}_{PSAPF} \begin{cases} > \overline{R}_{FCFS}, & C_D < 1, \\ = \overline{R}_{FCFS}, & C_D = 1, \\ < \overline{R}_{FCFS}, & C_D > 1. \end{cases} \tag{25}$$

These results are illustrated in Figure 9a, which plots $\overline{R}_{FCFS}/\overline{R}_{PSAPF}$ versus C_D for the H and M workloads given in Table 2. The response time ratios for the L workload are not shown because they lie very close to those for the H workload. We note from Figure 9a that C_D has a much stronger effect for the M workload as compared to the H and L workloads. This is because PSAPF is more highly differentiated from FCFS for the M workload in which there is a wider range of values for available parallelism as opposed to the H and L workloads, as shown by the cdfs in Table 2.

5.2.2 PSAPF versus FCFS: r=1

The performance of PSAPF can be expected to improve relative to FCFS as correlation increases, and thus $\overline{R}_{PSAPF}/\overline{R}_{FCFS}$ should be lower at $r = 1$ than at $r = 0$. Simulation experiments were run to obtain $\overline{R}_{FCFS}$ at $r = 1$ and C_v between 0 and 5 for the H, M, and L parallelism workloads. $\overline{R}_{PSAPF}$ is approximated using (22). Figure 9b plots the ratios $\overline{R}_{FCFS}/\overline{R}_{PSAPF}$ as a function of C_D for the three parallelism workloads. (For the L workload the results are shown up to $C_v = 3$ which corresponds to $C_D = 9.04$ using (3).) We observe that the mean response time ratios at r=1 are significantly lower than the ratios at r=0. The ratios decrease faster with increasing C_D, and also decrease more substantially for the



(a) $r = 0$ (b) $r = 1$

Figure 9: $\overline{R}_{FCFS}/\overline{R}_{PSAPF}$ versus C_D: γ^l

P=100

M workload compared with the H workload, and for the L workload compared with the M workload. The reason for the marked improvement in the L workload is that when $r = 1$ 90% of the jobs have lower mean demands than the other 10%. This differentiation in mean demands when $r = 1$ increases the performance differential between PSAPF and FCFS.

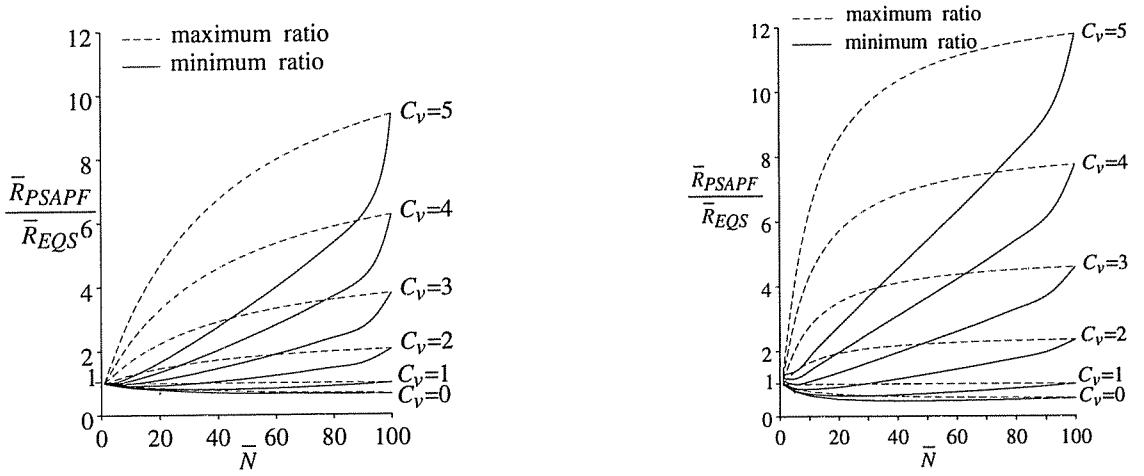
To summarize the comparison between PSAPF and FCFS, the results have shown that PSAPF performs better than FCFS for most of the parameter space. FCFS performs marginally better when $r = 0$ and $C_D < 1$. The quantitative results above were for the linear ERF. PSAPF should perform relatively even better if the ERF is sublinear, as explained above.

5.3 PSAPF versus EQS

Sections 5.1 and 5.2 respectively showed that in general EQS performs better than ASP and that PSAPF performs better than FCFS (unless r is low and $C_D < 1$). The EQS policy has high performance since it utilizes processors efficiently and its response time is insensitive to C_D (as shown by the approximate analysis in Section 3). The PSAPF policy has high performance for workloads with high correlation since it gives priority to jobs with small mean demand in these workloads. This section first compares PSAPF and EQS at $r = 1$ and $\gamma = \gamma^l$. Each of these parameter settings is more favorable to PSAPF relative to EQS. After

the comparison at $r - 1$, the policies are compared for $r < 1$.

When $r = 1$ and $\gamma = \gamma^l$, we have $\overline{S} = \overline{D}/\overline{N}$ (see the equation above (2)) and thus $S_n = 1/\overline{N}$. Since S_n is the key parallelism parameter for EQS, and $\overline{N}$ is uniquely determined from S_n when $r = 1$ and $\gamma = \gamma^l$, it follows that $\overline{N}$ is equivalently the key parallelism parameter for EQS under these workload conditions. That is, at $r = 1$ and $\gamma = \gamma^l$ $\overline{R}_{EQS}$ is approximately the same across all distributions of N with the same $\overline{N}$. This is not necessarily true for $\overline{R}_{PSAPF}$. Therefore, approximation (22) and nonlinear programming are used to obtain the minimum and maximum values of $\overline{R}_{PSAPF}$ across all distributions of N that have the same $\overline{N}$, and then use these values to determine the relative performance of PSAPF with respect to EQS. The details of the nonlinear programming solution method are given in Appendix C.



(a) $\rho = 0.7$ (b) $\rho = 0.9$

Figure 10: $\overline{R}_{PSAPF}/\overline{R}_{EQ}$ versus $\overline{N}$: $r = 1$, γ^l P=100, $\overline{D} = P$

Figure 10a plots the minimum and maximum of $\overline{R}_{PSAPF}/\overline{R}_{EQS}$ as a function of $\overline{N}$ for various C_v at $\rho = 0.7$ for a 100-processor system. We observe that the ratios increase with C_v since PSAPF is sensitive to C_v, whereas EQS is not. When $C_v \leq 1$ PSAPF performs better than EQS. However, the reverse is mostly true for $C_v = 2$ (in the range $\overline{N} \geq 30$) and is true for all cases with $C_v \geq 3$. We also observe that the minimum and maximum ratios increase with $\overline{N}$ for $C_v \geq 2$, due to the improvement in EQS performance with increase in average available parallelism when $r = 1$ [19]. Figure 10b plots similar ratios for $\rho = 0.9$ and we note that the difference between PSAPF and EQS increases with an increase in ρ.

Under sublinear execution rates the ratios of Figure 10 should be higher by virtue of Theorem 5.1. We can therefore conclude that for general workload conditions with $r = 1$, EQS outperforms PSAPF as long as $C_v > 2$.[14]

Coupling the results from Figure 10 for $r = 1$ with the results from [18] for $r = 0$ we have the following relationships between $\overline{R}_{EQS}$ and $\overline{R}_{PSAPF}$ at the extreme ends of correlation.

$$\overline{R}_{EQS}(r=0)\begin{cases} > \overline{R}_{PSAPF}(r=0), & 0 \le C_v < 1, \\ = \overline{R}_{PSAPF}(r=0), & C_v = 1, \\ < \overline{R}_{PSAPF}(r=0), & C_v > 1, \end{cases} \quad \text{and} \quad \overline{R}_{EQS}(r=1)\begin{cases} > \overline{R}_{PSAPF}(r=1), & 0 \le C_v < 1, \\ ? \ \overline{R}_{PSAPF}(r=1), & 1 \le C_v \le 2, \\ < \overline{R}_{PSAPF}(r=1), & C_v > 2, \end{cases} \tag{26}$$

where the question mark on the right reflects that the exact relationship between $\overline{R}_{EQS}(r = 1)$ and $\overline{R}_{PSAPF}(r = 1)$ is sensitive to the distribution of N. Note that in (26) we used the fact that at $r = 0$, we have $C_v = C_D$ as can be seen from (3). Now consider the case where $0 < r < 1$. The approximations for EQS(γ) and PSAPF(γ^l) (see (10) and (23)) for $0 < r < 1$ have the following forms

$$\begin{aligned} \overline{R}_{EQS} &\approx (1-r^2)\overline{R}_{EQS}(r=0) + r^2\overline{R}_{EQS}(r=1) \\ \overline{R}_{PSAPF} &\approx (1-r^2)\overline{R}_{PSAPF}(r=0) + r^2\overline{R}_{PSAPF}(r=1). \end{aligned}$$

Using these approximations and the relationships in (26) it follows that for general r, $\overline{R}_{EQS} > \overline{R}_{PSAPF}$ for $C_v < 1$ and $\overline{R}_{EQS} > \overline{R}_{PSAPF}$ for $C_v > 2$. For C_v between 1 and 2, the relative performance of EQS and PSAPF depends on the value of r and on the distribution of N. In general, workloads in general purpose computer systems have high variation in demand [31, pg16],[38][15] and in these systems we expect EQS to perform significantly better than PSAPF.

5.4 Generalization and Unification of Previous Work

The policy comparisons for the ASP, EQS, FCFS, and PSAPF policies in Section 5.1-5.3 enable us to delineate regions of the parameter space under which each policy performs best. The direct results derived assuming the linear ERF, together with the extensions for sublinear ERFs, leads to the delineation shown in Figure 11. The key determinants of *relative* policy performance are the axes of the figure. C_v has been shown

[14]Note that given C_v, C_N, and r, we can compute C_D using (3). For example, when $\overline{N} = 50$, $C_N \le 0.99$ from (4) and thus if $C_v = 2$, we get $C_D \le 3$.

[15]Note that we have also measured the coefficient of variation in service times on our local CM-5, which ranges from 2.8 to about 5, with the higher end being more typical.

to be a key parameter in all comparisons in this paper. Correlation between mean demand and available parallelism determines the relative performance between FCFS and PSAPF and between PSAPF and EQS. The ERF sublinearity affects relative policy performance since at very sublinear ERFs EQS will be perform best for all $C_v \geq 0$ and all $0 < r < 1$. Although S_n is a key determinant of absolute performance for ASP and EQS, this parameter is not shown in Figure 11, because the performance of EQS is better than that of ASP throughout the range of S_n.

We analytically derived results only for the topmost plane in the figure, where the ERF is linear, for which comparisons are favorable to ASP, FCFS, and PSAPF with respect to EQS. In this plane, the orderings between PSAPF and EQS, between FCFS and EQS at $r = 0$, and between FCFS and PSAPF at $r = 0$ were derived analytically. To supplement the results from analysis and complete the topmost plane of Figure 11 the following assumptions are required: (1) FCFS performs as well or better than ASP when $C_D = 0$ and S_n is close to one (i.e., in the narrow region where ASP performs marginally better than EQS), (2) $\overline{R}_{ASP}(C_D = 1)$ is a lower bound for its performance when $C_D > 1$, and (3) the simulation results for PSAPF versus FCFS when $r = 1$ (Section 5.3) hold qualitatively for all distributions of demand and parallelism. The value of r that delineates the boundary between FCFS and PSAPF in the topmost plane has not been precisely derived in this paper, and is thus indicated by the line break in the figure. Extending the results from the topmost plane to sublinear ERFs makes use of Theorem 5.1 which shows that EQS should perform relatively better than FCFS and PSAPF as the ERF sublinearity increases. The lack of precise values of γ that form the boundaries between these policies is also indicated by line breaks in the figure. Furthermore, the precise boundaries may depend not only on the specific degree of sublinearity but also on the specific ERF and the distribution of available parallelism. However, for $C_v \geq 1$ and $r = 0$ and $C_v \geq 2$ and $r = 1$ the result that EQS performs best holds for all distributions of N.

Due to the general workload assumptions in this paper the delineation of the design space generalizes and unifies previous results, as follows. First consider the line for $r = 0$ and $\gamma = \gamma^l$, and variable C_v.

- Two previous studies show that PSAPF, FCFS, and EQ have almost the same performance at $(C_v = 1, r = 0, \gamma^l)$ [14, 18]. This is shown in Figure 11 for FCFS and EQS[16] and approximation (25) shows that $\overline{R}_{PSAPF} \approx \overline{R}_{FCFS}$ under these conditions.
- For an uncorrelated workload ($r = 0$) with specific hyperexponential demand distributions ($C_v > 1$), specific distributions of N, and linear speedups, [14] shows that $\overline{R}_{EQ} < \overline{R}_{PSAPF}, \overline{R}_{FCFS}$. Figure 11 shows the same result for all distributions of demand and parallelism.

[16]Note that when γ is linear, all EQ policies have the same performance under the assumption of $E(j) = \gamma(j)$.

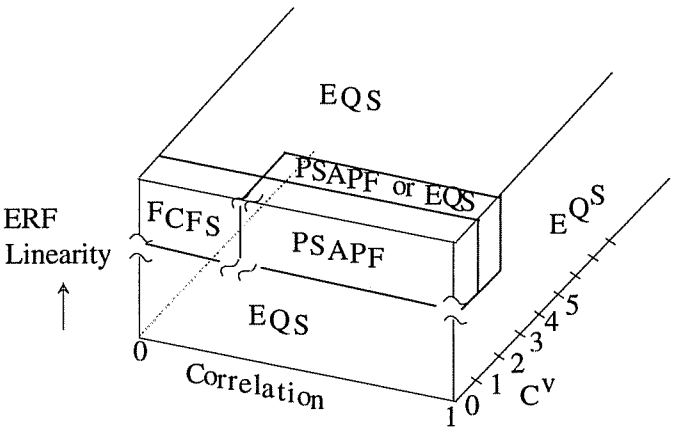


Figure 11: Summary of Policy Comparison Results

Now consider the line $r = 1$, $\gamma = \gamma^l$, and variable C_v.

- The results in [13] compare PSAPF, FCFS, EQ and several other policies for a workload with a fixed number of jobs having i.i.d. *exponential* task service times (for which $C_v < 1$ and $r = 1$) and linear task execution rates. PSAPF is shown to be the optimal policy for the given workload. This is consistent with Figure 11 which shows that *in general* for $C_v < 1$, $r = 1$, and $\gamma = \gamma^l$, PSAPF is optimal among ASP, FCFS, EQ, and PSAPF.
- The results in [14] show that $\overline{R}_{EQ} < \overline{R}_{PSAPF}$ for specific hyperexponential demands, specific parallelism distributions, full correlation ($r = 1$), and linear speedups. Additional simulation data in [12] for the same assumptions shows PSAPF to perform better than EQ if $C_v < 2$ and worse if $C_v > 2$. [12] also shows cases with C_v between 1 and 2 where PSAPF performs worse than EQ. Again these results are consistent with Figure 11.

Now consider $0 < r < 1$, which is the case in [33, 21, 22] where no quantitative measures of workload correlation are given. These studies show that for particular workloads with sublinear ERFs, EQS outperforms ASP under exponential per class demands ($C_v = 1$) [33] and under a specific mix of applications with $C_v > 1$ [21, 22]. The same result is shown in Section 5.1 and Figure 11 for all distributions of demand and parallelism.

Other results in the literature show that $\overline{R}_{PSAPF} < \overline{R}_{FCFS}$ for hyperexponential demands ($C_v > 1$), specific distributions of parallelism, and both $r = 0$ and $r = 1$ [15, 14]. The same result is shown in Section 5.2 for all distributions of demand with $C_v > 1$ and for all distributions of N. Finally, [37] shows FCFS to outperform Round Robin Process and Processor Sharing for i.i.d. generalized exponential task service times with coefficient of variation < 4. Note that for this model $r = 1$ and Figure 11 shows that if the C_v of the sum of task service times is low, then PSAPF performs better than FCFS whereas if C_v is high then EQS performs better.

6 Conclusions

This paper has compared the performance of four parallel processor allocation policies, ASP, EQS, FCFS, and PSAPF that were shown in previous literature to have high performance under specific workloads. The comparisons were made over a general workload model that includes general distribution of available job parallelism, controlled correlation between total job processing requirement (i.e. demand) and available parallelism, general distribution of demands for jobs with no correlation, and a general deterministic job execution rate function for all jobs. Under the assumption that jobs can dynamically and efficiently redistribute their work across the processors allocated to them, the mean response time of each policy was estimated using interpolation approximations for various regions of the parameter space. The mean response time formulas were validated against simulation and then used to obtain key determinants of policy performance. By using the key parameters to explore the design space, results that generalize and unify previous policy comparison results were obtained. The regions of the parameter space under which each policy performs best are delineated as in Figure 11.

The main results of this paper are as follows:

- Coefficient of variation in demand (C_D) can be critical in determining *relative* policy performance. This might be obvious from uniprocessor scheduling results, but most previous analyses and comparisons of parallel processor policies have assumed exponential demands or exponential task service times. This result shows that the assumption of a particular coefficient of variation in job demand can have a strong impact on policy comparisons.
- Execution rate sublinearity and correlation between demand and parallelism are also parameters that influence relative policy performance. While speedup curves have been explicitly considered and specified in previous studies, in many cases no measures of the assumed or inherent correlation are provided. This result shows that it is important to consider the correlation between demand and parallelism and its implications on relative policy performance.
- For a fixed set of jobs with a common nondecreasing and concave execution rate function the EQS policy achieves optimal processor utilization over all allocation policies.
- EQS substantially outperforms ASP for both uncorrelated as well as correlated workloads. This result is shown for more general demand and parallelism distributions than in previous studies.
- EQS outperforms PSAPF when C_D is moderate to high even when workload correlation is high. PSAPF has lower mean response time tha EQ only for highly correlated workloads at low to moderate values of C_D, and when execution rates are (close to) linear. These results are hold for all distributions of available parallelism and general distributions of demand.
- PSAPF outperforms FCFS for most of the parameter space. (FCFS has slightly lower mean response time when correlation is low, $C_D < 1$, and job execution rate is linear.)
- Since general purpose computer systems are likely to have high variation in job processing requirements, the EQS policy seems to be the best candidate for implementation among the policies considered in this paper.

The policies examined in this study are idealizations of practical processor allocation policies, since we have assumed zero scheduling and preemption overheads. The qualitative results should continue to hold if practical (approximate) implementations of these policies can ensure that the overheads are small compared to job service times. In this paper it was assumed that applications can dynamically and efficiently redistribute their work among allocated processors. In environments where this is not possible, based on the results of this paper, a natural candidate policy to consider is temporal equiallocation.

Acknowledgements

We are grateful to Steve Dirkse for implementing the nonlinear program in GAMS for minimizing mean response time under PSAPF, and to Michael Ferris for discussions on convexity of functions and convex sets.

References

[1] R. Agrawal, R. Mansharamani, and M. Vernon. Response time bounds for parallel processor allocation policies. Technical Report # 1152, Computer Sciences Department, University of Wisconsin-Madison, June 1993.

[2] M. Bazaraa, and C. Shetty. Nonlinear Programming: Theory and Algorithms. John Wiley & Sons, New York 1979.

[3] A. Bondi, and J. Buzen. The Response Times of Priority Classes under Preemptive Resume in M/G/m Queues. *Performance Evaluation Review 12*, 3 (August 1984), 195-201.

[4] A. Brooke, D. Kendrick, and A. Meerhaus. GAMS, a User's Guide. Scientific Press, Redwood City, CA, 1988.

[5] J. Buzen, and A. Bondi. The Response Time of Priority Classes under Preemptive Resume in M/M/m Queues. *Operations Research 31*, 2 (1983), 456-465.

[6] L. Dowdy. On the Partitioning of Multiprocessor Systems. Technical Report, Vanderbilt University, Nashville, TN, July 1988.

[7] D. Ghosal, G. Serazzi, and S. Tripathi. The Processor Working Set and Its Use in Scheduling Multiprocessor Systems. *IEEE Transactions on Software Engineering 17*, 5 (May 1991), 443-453.

[8] A. Gupta, A. Tucker, and L. Stevens. Making Effective Use of Shared Memory Multiprocessors: The Process Control Approach. Technical Report, Computer Sciences Department, Stanford University, Stanford, CA, July 1991.

[9] F. Kelly. *Reversibility and Stochastic Networks.* John Wiley & Sons, 1979.

[10] L. Kleinrock. *Queueing Systems, Vol I: Theory.* John Wiley & Sons, New York 1975.

[11] L. Kleinrock. *Queueing Systems, Vol II: Computer Applications.* John Wiley & Sons, New York 1976.

[12] S. Leutenegger. Issues in Multiprogrammed Multiprocessor Sharing. Ph.D. Thesis, Technical Report #954, Department of Computer Sciences, University of Wisconsin-Madison, August 1990.

[13] S. Leutenegger, and R. Nelson. Analysis of Spatial and Temporal Scheduling Policies for Semi-Static and Dynamic Multiprocessor Environments. Research Report-IBM T.J. Watson Research Center, Yorktown Heights, August 1991.

[14] S. Leutenegger, and M. Vernon. The Performance of Multiprogrammed Multiprocessor Scheduling Policies. *Performance Evaluation Review 18*, 1 (May 1990), 226-236.

[15] S. Majumdar, D. Eager, and R. Bunt. Scheduling in Multiprogrammed Parallel Systems. *Performance Evaluation Review 16*, 1 (May 1988), 104-113.

[16] S. Majumdar, D. Eager, and R. Bunt. Characterisation of Programs for Scheduling in Multiprogrammed Parallel Systems. *Performance Evaluation 13*, (1991), 109-130.

[17] R. Mansharamani. Efficient Analysis of Parallel Processor Scheduling Policies. Ph.D. Thesis, Computer Sciences Department, University of Wisconsin, Madison, WI, November 1993.

[18] R. Mansharamani, and M. Vernon. Approximate Analysis of Parallel Processor Allocation Policies. Technical Report 1187, Computer Sciences Department, University of Wisconsin, Madison, WI, November 1993.

[19] R. Mansharamani, and M. Vernon. Performance Analysis of the EQuipartitioning Parallel Processor Allocation Policy. *In preparation.*

[20] C. McCann, R. Vaswani, and J. Zahorjan. A Dynamic Processor Allocation Policy for Multiprogrammed, Shared Memory Multiprocessors. *ACM Transactions on Computer Systems 11*, 2 (May 1993), 146-178.

[21] V. Naik, S. Setia, and M. Squillante. Scheduling of Large Scientific Applications on Distributed Memory Multiprocessor Systems. *Proceedings of the 6th SIAM Conference on Parallel Processing for Scientific Computation.* IBM Research Report RC 18621, T. J. Watson Research Center, Yorktown Heights, Jan. 1993.

[22] V.Naik, S. Setia, and M. Squillante. Performance Analysis of Job Scheduling Policies in Parallel Supercomputing Environments. *Proceedings of Supercomputing'93*, November 1993.

[23] R. Nelson. A Performance Evaluation of a General Parallel Processing Model. *Proceedings of ACM SIGMETRICS Conference; Performance Evaluation Review 18*, 1 (May 1990), 13–26.

[24] R. Nelson. Matrix Geometric Solutions in Markov Models - A Mathematical Tutorial. Research Report - IBM T.J. Watson Research Center, Yorktown Heights, April 1991.

[25] R. Nelson, and D. Towsley. A Performance Evaluation of Several Priority Policies for Parallel Processing Systems. COINS Technical Report 91-32, Computer and Information Sciences, University of Massachusetts at Amherst, May 1991. (To appear in JACM.)

[26] R. Nelson, D. Towsley, and A. Tantawi. Performance Analysis of Parallel Processing Systems. *IEEE Transactions on Software Engineering 14*, 4 (Apr. 1988), 532–540.

[27] M. Neuts. Matrix-Geometric Solutions in Stochastic Models: An Algorithmic Approach. The John Hopkins University Press, 1981.

[28] A. Roberts, and D. Varberg. *Convex Functions.* Academic Press, New York, 1973.

[29] E. Rosti, E. Smirni, L. Dowdy, G. Serazzi, and B. Carlson. Robust Partitioning Policies of Multiprocessor Systems. Technical Report, Department of Computer Science, Vanderbilt University 1992. To appear, in *Performance Evaluation* (Special issue on the performance modeling of parallel processing systems).

[30] H. Sakasegawa. An approximation formula $L_q \doteq \alpha\rho^{\beta}/(1-\rho)$. *Annals of the Institute of Statistical Mathematics 29*, 1 (1977), 67-75.

[31] C. Sauer, and K. M. Chandy. *Computer System Performance Modeling.* Prentice-Hall, Englewood Cliffs, New Jersey, 1981.

[32] M. Seager, and J. Stichnoth. Simulating the Scheduling of Parallel Supercomputer Applications. Technical Report, User Systems Division, Lawrence Livermore National Laboratory, September 1989.

[33] S. Setia, and Tripathi. An Analysis of Several Processor Partitioning Policies for Parallel Computers. Technical Report CS-TR-2684, University of Maryland, College Park, MD, May 1991.

[34] S. Setia, and S. Tripathi. A Comparative Analysis of Static Processor Partitioning Policies for Parallel Computers. *Proceedings of the International Workshop on Modeling and Simulation of Computer and Telecommunication Systems (MASCOTS)*, 1993.

[35] K. Sevcik. Application scheduling and processor allocation in multiprogrammed parallel processing systems. To appear in a special issue of *Performance Evaluation.*

[36] N. Tabet-Aouel, and D. Kouvatsos. On an Approximation to the Mean Response Times of Priority Classes in a Stable G/G/c/PR Queue. *Jnl. of the Operational Research Society 43*, 3 (Mar 1992), 227-239.

[37] D. Towsley, C. Rommel, and J. Stankovic. Analysis of Fork-Join Program Response Times on Multiprocessors. *IEEE Transactions on Parallel and Distributed Systems 1*, 3 (July 1990), 286-303.

[38] K. Trivedi. *Probability and Statistics, with Reliability, Queueing and Computer Science Applications.* Prentice-Hall, 1982, pg. 130.

[39] A. Tucker and A. Gupta. Process Control and Scheduling Issues for Multiprogrammed Shared-Memory Multiprocessors. *Proc. of the 12th ACM Symp. on Operating System Principles*, Dec. 1989, 159-166.

[40] R. Vaswani and J. Zahorjan. The Implications of Cache Affinity on Processor Scheduling for Multiprogrammed, Shared Memory Multiprocessors. *Proc. of the 13th ACM Symposium on Operating System Principles*, October 1991, 26-40.

[41] J. Zahorjan, and C. McCann. Processor Scheduling in Shared Memory Multiprocessors. *Performance Evaluation Review 18*, 1 (May 1990), 214-225.

[42] S. Zhou, and T. Brecht. Processor-pool-based Scheduling for Large-Scale NUMA Multiprocessors. *Performance Evaluation Review 19*, 1 (May 1991), 133-142.

Appendix

A Derivation of $\overline{R}_{PSAPF}$ approximation for $r = 1$

In this section we derive (21) using the per class mean response time approximation for an M/G/c PR queue given in [36]. That is, we show that,

$$\overline{R}_{\Gamma_k,C_k} \approx c\,\overline{x}_k + \frac{1}{p_k}\left(\sum_{i=1}^{k-1} g_i\right)\left(\sigma_k^{\sqrt{2(c+1)}-2} - \sigma_{k-1}^{\sqrt{2(c+1)}-2}\right) + \frac{1}{p_k} g_k \sigma_k^{\sqrt{2(c+1)}-2}, \quad p_k > 0, \tag{27}$$

where $c = P/k$, $\overline{x}_i = (\overline{D}i)/(\overline{N}P)$, $\sigma_i = \lambda \sum_{j=1}^{i} p_i \overline{x}_i$, and

$$g_i = p_i \frac{\sigma_{i-1}}{1-\sigma_{i-1}}\overline{x}_i + \frac{\lambda p_i \sum_{j=1}^{i}\{p_j(1+C_v^2)\overline{x}_j^2\}}{2(1-\sigma_{i-1})(1-\sigma_i)}.$$

Recall that $\overline{R}_{\Gamma_k,C_k}$ in Section 3.5.1 was shown equal to the mean response time of the k^{th} priority class in an $M/G/c$ PR queue with k priority classes. To estimate $\overline{R}_{\Gamma_k,C_k}$ we use Tabetaeoul and Kouvatsos' heuristic [36] for per class mean response times of an $M/G/c$ PR queue. To begin with, $\overline{R}_{\Gamma_k,C_k}$ can be expressed in terms of the overall mean response times for the first i classes, $\overline{T}^i_{M/G/c\ PR}$, for $i = k-1, k$. That is,

$$\overline{R}_{\Gamma_k,C_k} = \frac{\Lambda_k \overline{T}^k_{M/G/c\ PR} - \Lambda_{k-1}\overline{T}^{k-1}_{M/G/c\ PR}}{\lambda_k}, \tag{28}$$

where $\lambda_k = \lambda p_k$, and $\Lambda_k = \sum_{i=1}^{k} \lambda_i$. Let $\overline{z}_i$ denote the service time of class i in the M/G/c PR queue, for $i = 1, \ldots, k$, and let $\overline{y}_i$ denote the overall mean service time of the first i classes, that is $\overline{y}_i = \frac{1}{\Lambda_i}\sum_{j=1}^{i} \lambda_i \overline{z}_i$ for $i = 1, \ldots, k$. Define $\overline{X}^k_{M/G/c\ PR} \equiv \overline{T}^k_{M/G/c\ PR} - \overline{y}_k$. Tabetaeoul and Kouvatsos [36] proposed the following heuristic to estimate $\overline{X}^k_{M/G/c\ PR}$.

$$\overline{X}^k_{M/G/c\ PR} \approx \overline{X}^k_{M/G/1_c\ PR} \cdot \frac{\overline{X}^k_{M/G/c}}{\overline{X}^k_{M/G/1_c}}, \tag{29}$$

where $\overline{X}^k_{M/G/1_c\ PR} \equiv \overline{T}^k_{M/G/1_c\ PR} - \overline{y}_k/c$, $\overline{T}^k_{M/G/1_c\ PR}$ being the overall mean response time of the first k classes in an $M/G/1_c$ PR queue that is obtained by replacing the c servers of the M/G/c queue by a

single server c times faster. (The $M/G/1_c$ PR system has the same job priorities, service demands, and arrival rates as the M/G/c PR system.) Similarly, $\overline{X}^k_{M/G/c}$ is the overall mean waiting time in an M/G/c system that has the same workload as the M/G/c PR system (i.e., k classes) and $\overline{X}^k_{M/G/1_c}$ is the overall mean waiting time in an equivalent $M/G/1_c$ system.

We first provide closed form expressions for $\overline{X}_{M/G/c}$ and $\overline{X}_{M/G/1_c}$ and then for $\overline{X}_{M/G/1_c\ PR}$ so that we can get an expression for $\overline{X}^k_{M/G/c\ PR}$ using (29). Using Sakasegawa's approximation for GI/G/c FCFS queues [30], we obtain

$$\overline{X}^k_{M/G/c} \approx \frac{\sigma_k^{\sqrt{2(c+1)}}(1+CV_k^2)}{2\Lambda_k(1-\sigma_k)},$$

where CV_k is the overall coefficient of variation in the service requirement of the k classes, and $\sigma_k = \sum_{i=1}^k \lambda_i \overline{z}_i/c$. Using the analysis in [10] for the M/G/1 queue we get,

$$\overline{X}^k_{M/G/1_c} = \frac{\sigma_k^2(1+CV_k^2)}{2\Lambda_k(1-\sigma_k)},$$

and as a result,

$$\frac{\overline{X}^k_{M/G/c}}{\overline{X}^k_{M/G/1_c}} \approx \sigma_k^{\sqrt{2(c+1)}-2}. \tag{30}$$

From the analysis in [11] we have the following expression for $\overline{X}^k_{M/G/1_c\ PR} \equiv \overline{T}^k_{M/G/1_c\ PR} - \overline{y}_k/c$.

$$\overline{X}^k_{M/G/1_c\ PR} = \frac{1}{\Lambda_k}\sum_{i=1}^k \lambda_i \left\{ \frac{\sigma_{i-1}}{(1-\sigma_{i-1})} \cdot \frac{\overline{z}_i}{c} + \frac{\sum_{j=1}^i \lambda_j (1+C_v^2)\frac{\overline{z}_j^2}{c^2}}{2(1-\sigma_{i-1})(1-\sigma_i)} \right\},$$

where C_v is the coefficient of variation of service requirement of class j, for $j = 1, \ldots, k$.

Substituting the above expression for $\overline{X}^k_{M/G/1_c\ PR}$ along with (30) into (29) we obtain

$$\overline{X}^k_{M/G/c\ PR} \approx \frac{1}{\Lambda_k}\left[\sum_{i=1}^k \lambda_i \left\{ \frac{\sigma_{i-1}}{1-\sigma_{i-1}} \cdot \frac{\overline{z}_i}{c} + \frac{\sum_{j=1}^i \lambda_j (1+C_v^2)\frac{\overline{z}_j^2}{c^2}}{2(1-\sigma_{i-1})(1-\sigma_i)} \right\}\right] \sigma_k^{\sqrt{2(c+1)}-2}.$$

Using $\overline{T}^k_{M/G/c\ PR} = \overline{X}^k_{M/G/c\ PR} + \overline{y}_k/c$, substituting into (28), and simplifying we get

$$\overline{R}_{\Gamma_k, C_k} \approx \overline{z}_k + \frac{1}{\lambda_k}\lambda \left(\sum_{i=1}^{k-1} g_i\right)\left(\sigma_k^{\sqrt{2(c+1)}-2} - \sigma_{k-1}^{\sqrt{2(c+1)}-2}\right) + \frac{1}{p_k}\lambda\, g_k \sigma_k^{\sqrt{2(c+1)}-2}, \quad p_k > 0,$$

where

$$g_i = p_i \frac{\sigma_{i-1}}{1-\sigma_{i-1}} \overline{z}_i/c \; + \; \frac{\lambda p_i \sum_{j=1}^{i} \{p_j(1+C_v^2)\overline{z}_j^2/c^2\}}{2(1-\sigma_{i-1})(1-\sigma_i)}.$$

We used $\lambda_i = \lambda p_i$ to obtain g_i. Substituting $\overline{x}_i = \overline{z}_i/c$ into the above expressions for $\overline{R}_{\Gamma_k,C_k}$ and g_i we obtain (27) as required. Note that $\overline{z}_i = \overline{D}_i/k = (\overline{D}i)/(\overline{N}k)$, and since $c = P/k$, we get $\overline{x}_i = (\overline{D}i)/(\overline{N}P)$, for $i = 1, \ldots, k$.

B Proof of Theorem 4.1

Theorem 5.1 *Consider a set of K jobs with available parallelisms $(n_1, \ldots, n_K)$. Let Ψ be a processor allocation policy that allocates a_i^{Ψ} processors to job i, for $i = 1, \ldots, K$. Then for a workload ERF γ that is concave and nondecreasing, and for $E(j) = \gamma(j)$, i.e., jobs dynamically and efficiently redistribute their work,*

$$\sum_{i=1}^{K} E(a_i^{EQS}) \; \geq \; \sum_{i=1}^{K} E(a_i^{\Psi}), \quad \textit{for any processor allocation policy } \Psi.$$

Proof. Without loss of generality assume that $n_1 \leq n_2 \leq \ldots \leq n_K$. Throughout the proof we use the following two properties:

- $\sum_{i=1}^{K} a_i^{\Psi} \leq P$.
- $a_i^{EQS} \leq n_i$, but a_i^{Ψ} can be $\geq n_i$ if Ψ is not processor conserving, $i = 1, \ldots, K$.

We divide the proof into three cases.

Case 1: $\sum_{i=1}^{K} n_i \leq P$.

We have $a_i^{EQS} = n_i$, $i = 1, \ldots, K$ and thus,

$$\sum_{i=1}^{K} E(a_i^{EQS}) = \sum_{i=1}^{K} E(n_i) \geq \sum_{i=1}^{K} E(a_i^{\Psi}),$$

where the last inequality follows because E is nondecreasing and $E(x) = E(N)$ for $x > N$.

Case 2: $n_1 \geq P/K$, i.e., all jobs under EQS get P/K processors each.

By definition, $a_i^{EQS} = P/K$. Therefore,

$$\sum_{i=1}^{K} E(a_i^{EQS}) = K\gamma\left(\frac{P}{K}\right) \geq K\gamma\left(\frac{\sum_{i=1}^{K} a_i^{\Psi}}{K}\right) \geq \sum_{i=1}^{K} \gamma(a_i^{\Psi}) \geq \sum_{i=1}^{K} E(a_i^{\Psi}),$$

where we have used the fact that γ is concave and nondecreasing and $E(x) \leq \gamma(x)$ if $x > N$.

Case 3: $n_1 < P/K$.

Under EQS jobs with low available parallelism are allocated as many processors as their available parallelisms and the remaining jobs get the resultant equipartition number. Let the first J jobs under EQS get as many processors as their available parallelisms. That is, $a_i^{EQS} = n_i$, for $i = 1, \ldots, J$ and $a_i^{EQS} = (P - \sum_{\ell=1}^{J} n_\ell)/(K - J)$ for $i = J+1, \ldots, K$, where $a_j^{EQS} \leq a_k^{EQS}$ for $j \in \{1, \ldots, J\}$ and $k \in \{J+1, \ldots, K\}$. We now have

$$\sum_{i=1}^{K} E(a_i^{EQS}) = \sum_{i=1}^{J} \gamma(n_i) + (K-J)\gamma\left(\frac{P - \sum_{i=1}^{J} n_i}{K-J}\right). \tag{31}$$

Let $u_i^{\Psi} = \min(a_i^{\Psi}, n_i)$ be the *useful* processor allocation under policy Ψ. As a result for policy Ψ we have

$$\sum_{i=1}^{K} E(a_i^{\Psi}) = \sum_{i=1}^{J} \gamma(u_i^{\Psi}) + \sum_{i=J+1}^{K} \gamma(u_i^{\Psi}) \leq \sum_{i=1}^{J} \gamma(u_i^{\Psi}) + (K-J)\gamma\left(\frac{P - \sum_{i=1}^{J} u_i^{\Psi}}{K-J}\right), \tag{32}$$

where the last inequality follows due to concavity of γ. Using (31) and (32), to prove the theorem it suffices to show that

$$\sum_{i=1}^{J} \gamma(n_i) + (K-J)\gamma\left(\frac{P - \sum_{i=1}^{J} n_i}{K-J}\right) \geq \sum_{i=1}^{J} \gamma(u_i^{\Psi}) + (K-J)\gamma\left(\frac{P - \sum_{i=1}^{J} u_i^{\Psi}}{K-J}\right). \tag{33}$$

To complete the proof we make use of the following property of concave functions. For a concave function f

$$\frac{f(x_2) - f(x_1)}{x_2 - x_1} \geq \frac{f(x_4) - f(x_3)}{x_4 - x_3}, \quad \text{where } x_1 \leq x_2 \leq x_3 \leq x_4. \tag{34}$$

This property is best illustrated by Figure 12 where slope of line AB $\geq$ slope of line CD. Since $u_i^{\Psi} \leq n_i$, for $i = 1, \ldots, J$ we have $(P - \sum_{i=1}^{J} u_i^{\Psi})/(K-J) \geq (P - \sum_{i=1}^{J} n_i)/(K-J)$. Therefore,

$$u_i^{\Psi} \leq n_i \leq \frac{P - \sum_{i=1}^{J} n_i}{K-J} \leq \frac{P - \sum_{i=1}^{J} u_i^{\Psi}}{K-J},$$

where the second inequality is a consequence of $a_j^{EQS} \leq a_k^{EQS}$ for $j \in \{1, \ldots, J\}$ and $k \in \{J+1, \ldots, K\}$. Applying (34) we obtain,

$$\frac{\gamma(n_i) - \gamma(u_i^{\Psi})}{n_i - u_i^{\Psi}} \geq \frac{\gamma\left(\frac{P-\sum_{\ell=1}^{J} u_\ell^{\Psi}}{K-J}\right) - \gamma\left(\frac{P-\sum_{\ell=1}^{J} n_\ell}{K-J}\right)}{\left(\frac{P-\sum_{\ell=1}^{J} u_\ell^{\Psi}}{K-J}\right) - \left(\frac{P-\sum_{\ell=1}^{J} n_\ell}{K-J}\right)} = \frac{\gamma\left(\frac{P-\sum_{\ell=1}^{J} u_\ell^{\Psi}}{K-J}\right) - \gamma\left(\frac{P-\sum_{\ell=1}^{J} n_\ell}{K-J}\right)}{\frac{\sum_{\ell=1}^{J}(n_\ell - u_\ell^{\Psi})}{K-J}}, \quad i = 1, \ldots, J.$$

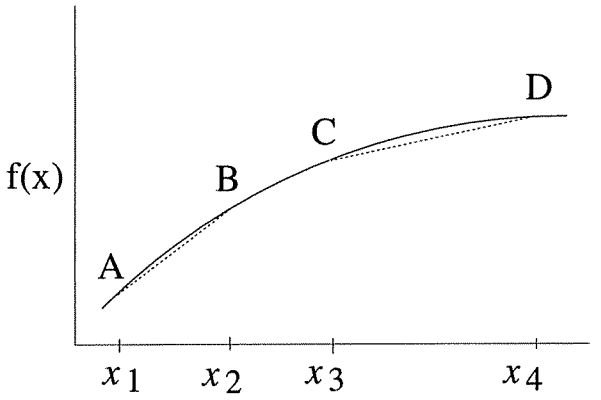


Figure 12: A Property of Concave Functions

As a result,

$$\gamma(n_i) - \gamma(u_i^{\Psi}) \geq (n_i - u_i^{\Psi}) \cdot \left[(K-J) \cdot \frac{\gamma\left(\frac{P-\sum_{\ell=1}^{J} u_\ell^{\Psi}}{K-J}\right) - \gamma\left(\frac{P-\sum_{\ell=1}^{J} n_\ell}{K-J}\right)}{\sum_{\ell=1}^{J}(n_\ell - u_\ell^{\Psi})} \right], \quad \text{for } i = 1, \ldots, J.$$

Summing both sides from $i = 1$ to J we get,

$$\sum_{i=1}^{J} \gamma(n_i) - \sum_{i=1}^{J} \gamma(u_i^{\Psi}) \geq (K-J) \cdot \left\{ \gamma\left(\frac{P - \sum_{\ell=1}^{J} u_\ell^{\Psi}}{K-J}\right) - \gamma\left(\frac{P - \sum_{\ell=1}^{J} n_\ell}{K-J}\right) \right\}.$$

Rearranging terms,

$$\sum_{i=1}^{J} \gamma(n_i) + (K-J)\,\gamma\left(\frac{P - \sum_{i=1}^{J} n_i}{K-J}\right) \geq \sum_{i=1}^{J} \gamma(u_i^{\Psi}) + (K-J)\,\gamma\left(\frac{P - \sum_{i=1}^{J} u_i^{\Psi}}{K-J}\right),$$

which is what we set out to prove (see (33)).

∎

C Obtaining Minimum and Maximum values of $\overline{R}_{PSAPF}$ at $r = 1$ and $\gamma = \gamma^l$

This appendix provides details of the algorithm used to find the minimum and maximum $\overline{R}_{PSAPF}$ at $r = 1$ and $\gamma = \gamma^l$ across all distributions of N that have the same $\overline{N}$. Approximation (22) estimates $\overline{R}_{PSAPF}$ under $r = 1$ and $\gamma = \gamma^l$. The objective is to minimize or maximize $\overline{R}_{PSAPF}$ over all pmf's $\underline{p}$ subject to the constraint that $\sum_{i=1}^{P} ip_i = \overline{N}$. It is easy to verify that the set of pmf's that satisfies this equality constraint is a convex set[17]. We henceforth denote this convex set by Ω. From [2] we note the following:

(1) Any local minimum of a convex function over a convex set is also a global minimum.

(2) If a convex function has a maximum over a convex set S, then the maximum is achieved at an extreme point of S, where an extreme point is a point that does not lie strictly within the line segment connecting two other points of the set.

If we can show that $\overline{R}_{PSAPF}$ is convex in $\underline{p}$ over the set Ω, our task of finding the global minimum and maximum values of $\overline{R}_{PSAPF}$ over Ω will be considerably simplified. (Note that in general there is no known algorithm that obtains the global minimum and maximum for an arbitrary nonlinear function.) We have been unable to rigorously prove that $\overline{R}_{PSAPF}$ is convex as desired, but we have empirically verified this property by selecting random pairs of points in Ω and verifying that the line segment connecting the mean response time values between each pair lies above the mean response time function. We have also plotted the shape of $\overline{R}_{PSAPF}$ for 2 and 3 dimensional problem sizes and verified it to be convex in Ω. We shall therefore assume that $\overline{R}_{PSAPF}$ is convex in Ω and use properties (1) and (2) from above.

The minimum values of $\overline{R}_{PSAPF}$ were obtained by writing a nonlinear program in GAMS [4], and running the program over the input values of λ, C_v, and $\overline{N}$ required for Figure 10. We specified several different initial feasible points $\underline{p} \in \Omega$ to the GAMS program and always obtained the same value for the minimum

[17] A nonempty set S in IR^n is said to be convex if the line segment joining any two points of the set also belongs to the set, i.e., if $\overline{x}_1$ and $\overline{x}_2$ are in S, then $\lambda\overline{x}_1 + (1-\lambda)\overline{x}_2$ is also in S for all λ between 0 and 1 [2].

(within the limits of numerical error), thus strengthening our belief that $\overline{R}_{PSAPF}$ is convex in Ω. To obtain the maximum values of $\overline{R}_{PSAPF}$ we first computed the extreme points in Ω. It can be verified that an extreme point in Ω is obtained by considering only two nonzero values in the pmf $\underline{p}$ and attaching suitable weights to them so that the mean is $\overline{N}$.[18] That only two nonzero values are needed results from the fact that there are only two equality constraints, the first $\sum p_i = 1$ and the second $\sum i p_i = \overline{N}$. Once the extreme points in Ω were obtained we then computed $\overline{R}_{PSAPF}$ at these points and selected the maximum value.

[18]In the special case where $\overline{N}$ is an integer, the point ($p_{\overline{N}} = 1$, $p_i = 0$ for $i \neq \overline{N}$) will also be an extreme point.