Size-based Scheduling with Differentiated Services to Improve Response Time of Highly Varying Flow Sizes

Idris A. Rai, Guillaume Urvoy-Keller, Ernst W. Biersack
Institut Eurecom
2229, Route des Crêtes
06904 Sophia-Antipolis, France
email:{rai,urvoy,erbi}@eurecom.fr

Abstract

The sizes of Internet objects are known to be highly varying. We evaluate an M/G/1 queue under *foreground background N* (FB_N) scheduling policy for job size distributions with varying *coefficient of variability* (CoV) to analyze the impact of variability of job sizes to the performance of the policy. We find that FB_N is very efficient in reducing the response time and minimizing the number of jobs that are penalized (i.e., have a higher response time under FB_N than under *processor sharing* (PS)) when job sizes have a high CoV. We also propose and analyze variants of FB_N called *fixed priority* FB_N ($FP-FB_N$) and *differential* FB_N ($DF-FB_N$), which introduce service differentiation by classifying jobs into *high priority* and *low priority* and then servicing the high priority before low priority jobs in an FB_N related order. The numerical analysis conducted for highly varying job sizes reveals that $FP-FB_N$ achieves a perfect service differentiation at the expense of a high penalty for the low priority small jobs. While $DF-FB_N$ offers acceptable service differentiation, it does not penalize small jobs with low priority at all. Moreover, $FP-FB_N$ and $DF-FB_N$ can guarantee the service of high priority jobs even under overload.

Keywords: highly varying traffic, heavy-tail, foreground background, size-based scheduling, service differentiation.

1 Introduction

Delay is a key metric for the quality-of-service perceived by end users. In today's Internet, packets experience delay due to transmission, propagation through the medium, and queueing in routers. The sum of these delay components, when accounted on end-to-end basis, is referred to as the *response time*. Research has shown that the queuing delay makes up a significant fraction of the response time, particularly at high load. Scheduling policies used at routers have a significant impact on the queueing delay, and their performances depend on the traffic variability.

Evidence of high variability in the Internet traffic has been widely observed with respect to the sizes of data objects in computer systems. In particular, data files transferred through the Internet [13, 2], files stored in Web servers [5, 2], and flows service times in Internet [7, 12]. The high variability attribute of the Internet traffic is sometimes referred to as heavy tail property [9, 6]. In this paper, the high variability in the Internet is characterized as the Internet traffic consisting of many small jobs with small sizes mixed with a few, very large jobs. In particular, we consider a flow size distribution to be highly varying if its *coefficient of variability* (CoV) is high, and about half of the total system load is due to very few largest flows. The CoV, which is defined as the ratio of the standard deviation to the mean of a distribution, is a useful metric to determine the variability of a distribution. We analyze a size-based scheduling policy called *Foreground Background N* (FB_N) in an M/G/1 system to see the impact of high variability in the response time. In this paper, we refer to response time as the overall time a job spends in the M/G/1 system. The term job is used to denote any entity that represents information in a network, e.g., a flow, a connection, or a session.

Policies that favor short jobs have been known to minimize the mean response time. But they have been long known in queueing theory to highly penalize large jobs. This however, is not necessarily true when the job size distribution has a high variance. The *shortest remaining processing time* algorithm (SRPT) favors small jobs by giving priority to short jobs or jobs with shortest remaining time. It is proven in [1] that for highly varying job sizes, even the largest jobs are either not penalized at all or see a negligible penalty. Similarly, we demonstrated in [15] that the *foreground background infinity* (FB_∞) scheduling policy, a policy that also favors small jobs over large ones by giving service to a job or jobs in the system that have received the least service among all jobs in the system [4], slightly increases the response time of the largest jobs with a significant reduction in the response times of the small jobs for job size distributions with high CoV.

SRPT and FB_∞ are not generic as they cannot be implemented everywhere in the network. SRPT is limited to network environments where job sizes are known [19]. An implementation of FB_∞ is not limited to the knowledge of job sizes, but it requires an infinite number of queues to ideally realize [18, 4]. Although we showed in [16] that it is feasible to implement FB_∞ by allocating a separate queue to each flow, this implementation is not scalable in routers with a large number of flows such as core routers. In this paper, we analyze a variant of FB_∞ called FB_N, which is a size-based scheduling policy that also favors small jobs like FB_∞, but maintains a finite number of N queues.

The N queues in FB_N are classified as $(N-1)$ *foreground queues* and one *background queue*. In FB_N, all queues share a single server according to their priorities. Let the first foreground queue be indexed 1 and the background queue indexed N, then a job at the head of queue j receives service if all queues $i \in \{1, 2, ..., N\}, i < j$ are empty. In FB_N, all jobs arrive at the first (highest priority) foreground queue where they receive, in FIFO order, a fixed amount of service called quantum. A job then either leaves the system if it has completed its service or is relayed to the subsequent queue. From each foreground queue, the job receives a service of one quantum if the previous queues are empty, until it is completely served. Jobs that complete their service in one of the foreground queues are called *foreground jobs*, and jobs that have not completed their service in one of the foreground queues are called *background jobs*. Background jobs are also serviced in FIFO order. A background job returns back at the *head* of the background queue after each quantum of service, where it is immediately taken back to service if all foreground queues are still empty. The process continues until the jobs complete service. Quantum values in FB_N can be the same for all queues or can be different for each queue.

The service received by foreground jobs under FB_N is the same as their services under FB_∞. Hence, the benefits of FB_∞ in minimizing the mean response time of jobs with high CoV can be reaped from all network routers by deploying a scalable FB_N policy instead. We present a numerical analysis of FB_N in Section 4, where we compare its offered *slowdown* to the slowdown offered by FB_∞ and processor sharing PS. Slowdown is the normalized response time, i.e., the ratio of the mean response time of a job to its size. The slowdown metric is important to analyze the fairness of a scheduling policy when compared to the slowdown of a fair policy, like processor sharing, which offers the same mean slowdown to all jobs. The results in this paper show that FB_N favors more foreground jobs for job size distribution with a high CoV than with low CoV, and the percentage of jobs that are *penalized* under FB_N is less for a job size distribution with a high CoV than with low CoV, particularly at load close to 1. We say a job under a scheduling policy is penalized if it has higher slowdown under that policy than under PS.

While the background jobs that are penalized under FB_N comprise a very tiny percentage of jobs in case of job size distribution with high CoV, less than 1%, this penalty is not acceptable by jobs or users that are classified as important. FB_N however, cannot differentiate the service of jobs using any attribute other than their sizes. Hence, it cannot guarantee the service quality of important jobs or users. Moreover, FB_N cannot service background jobs under overload. In Section 6, we propose and evaluate variants of FB_N that differentiate the service among jobs by classifying them into *high priority jobs* and *low priority jobs* based on desired attributes. The objective is to guarantee the service quality of the high priority jobs.

We first propose *fixed priority* FB_N $(FP-FB_N)$ scheduling policy, which serves jobs of each priority in a separate FB_N system such that low priority jobs receive service only if there is no high priority job in the corresponding FB_N system. In $FP-FB_N$, the service of the priority job depends only on the load that they constitute rather than the total system load. Therefore, $FP-FB_N$ can guarantee the service of the high priority jobs even under overload, as long as they constitute load of less than 1, which should be the case in practice. But $FP-FB_N$ significantly improves the service of high priority jobs at the expense of high response time to low priority small jobs. However, the mean slowdown of all small jobs under FB_N (without service differentiation) is significantly low. Since these jobs constitute large percentage of all jobs in highly varying job sizes, penalizing them, as in $FP-FB_N$, degrades the overall system performance. We propose *differential* FB_N $(DF-FB_N)$ policy, which also significantly reduces the response time of the high priority background jobs while maintaining the mean slowdown of low priority small jobs as low as under FB_N. This is achieved by differentiating only the service of the background jobs such that the low priority background jobs receive service in the background queue only when there are no high priority background (and foreground) jobs in the system. Similarly, the high priority background jobs under $DF-FB_N$ can receive service even under overload.

The rest of the paper is organized as follows: in the next section, we discuss the related previous work. Foreground background scheduling and mathematical expressions of the performance metrics are presented in Section 3. In Section 5, we discuss FB_N under overload. We analyze and evaluate differential and priority FB_N's in Section 6 and conclude the paper in Section 7.

2 Previous Work

There is a significant amount of research that makes use of high variability attribute of Internet job sizes to improve the quality of service of the jobs. This research focuses on different issues such as reducing delay in networks, routing, or bandwidth sharing.

In [1], SRPT is proposed to improve the performance of HTTP requests in Web servers. Experiments with the kernel level implementation of SRPT were executed in a

LAN and a WAN environments. The results show that large requests are negligibly penalized in case of high CoV job sizes under SRPT. Another attempt that uses SRPT to perform size based scheduling is [8], where connection scheduling is done in a Web server. The results show an improvement in the mean response time by a factor close to 4. Another paper that considers size-based scheduling using SRPT in Web servers is [3]. The authors suggest that using SRPT causes large files to have an arbitrarily high maximum slowdown. However, they assumed a worst-case adversarial sequence of Web requests in the paper. Roberts et. al. [17] suggest that SRPT may be beneficial in sharing bandwidth on a link in case of highly varying job sizes. In [16], we proposed a flow level implementation of FB_∞ for access routers of a virtual private network (VPN) with provisioned core services, where the reduction in response time at the edge directly reduces the end-to-end delay. We demonstrated that the feasibility of the implementation in edge devices relies on a moderate number of flows.

Shaikh et. al. [20] propose a load balancing routing technique that is based on flow sizes. The authors show that load sensitive routing can be efficiently made stable if applied to only long-lived flows. The success of this load-balancing technique depends on the variability of flow sizes, since for highly varying flow sizes dynamically routing less than 1% of flows means dynamically routing more than half of the load.

The work of this paper was motivated by our previous work in [15] where we analyzed FB_∞ and the fact that FB_N services foreground jobs like FB_∞. We showed in [15] that FB_∞ minimizes the mean slowdown of small jobs while maintaining a reasonable penalty for large jobs in terms of the mean slowdowns when the job size distribution has a high variance. Also, we demonstrated that FB_∞ offers a lower mean response time than FIFO for job size distributions with high CoV. Moreover, the stability of FB_∞ under overload conditions proved in [15] is an important advantage of FB_N over traditional FIFO and PS policies. The results in [14] and [15] show that while SRPT is optimal, FB_N and FB_∞ are quite close to SRPT for job sizes with a high CoV.

3 Foreground Background Scheduling

In this section, we discuss the mathematical preliminaries for foreground background scheduling and present expressions for the performance metrics considered in the paper. These performance metrics are conditional mean response time defined as $E[T(x)] \triangleq E[(T|X = x)]$ and conditional mean slowdown ($E[S(x)]$), which is defined as $E[S(x)] \triangleq \frac{E[T(x)]}{x}$, where X is a random variable representing the job size x. It follows that for any two scheduling policies A and B we have $\frac{E[T(x)]_A}{E[T(x)]_B} = \frac{E[T(x)]_A/x}{E[T(x)]_B/x} = \frac{E[S(x)]_A}{E[S(x)]_B}$.

3.1 Preliminaries

Let the average job arrival rate be λ. Assume that the probability density function of the job size X is $f(x)$. Given the cumulative distribution function $F(x) \triangleq \int_0^x f(t)dt$, we denote the survivor function of X as $F^c(x) \triangleq 1 - F(x)$. Let us consider the truncated distribution of X at x. The p.d.f of this truncated distribution is given as:

$$f_x(y) \triangleq \begin{cases} f(y) & \text{if } y < x \\ F^c(x) & \text{if } y = x \\ 0 & \text{otherwise.} \end{cases}$$

The moments of a random variable X_x of the truncated p.d.f $f_x(y)$ are defined as:

$$\overline{x_x^n} \triangleq \int_0^x y^n dF(y) + x^n F^c(x). \quad (1)$$

Integrating the first term of Equation (1) by parts, we obtain $\overline{x_x^n} = n\int_o^x y^{(n-1)}F^c(y)dy$. Equation (1) shows the moments that account for the contribution of all jobs to the response time of the job of size x (see [10], pp. 173), and we note that $\overline{x_\infty^n}$ are the moments of the original distribution. The load associated with all jobs of sizes less than or equal to x for the truncated distribution is $\rho_x \triangleq \lambda\overline{x}_x$ and $\rho_\infty = \rho$ is the total load.

Foreground background (FB) scheduling is a multilevel queueing policy that services jobs based on their sizes. The service time received by a job in a queue j of a foreground background scheduling policy is called a quantum and has a value of s_j. If quantum values of all queues of FB are infinitesimally small, the FB policy is called *processor sharing FB*. An FB scheduling algorithm that requires an infinite number of multilevel queues to realize is denoted as FB_∞. The formula for $E[T(x)]_{FB_\infty}$ for processor sharing model of FB_∞ is given in [18, 4] as:

$$E[T(x)]_{FB_\infty} = \frac{\lambda\overline{x_x^2}}{2(1-\rho_x)^2} + \frac{x}{(1-\rho_x)}. \quad (2)$$

FB_∞ is the most suitable size based policy that minimizes the mean slowdown of jobs without prior knowledge of their sizes [10], but the fact that it requires an infinite number of queues is a major drawback in its implementation. Also, the quantum values in a practical scheduling policy are nonzero. Thus, we analyze another foreground background scheduling policy called FB_N, which maintains a fixed number of N queues and uses fixed size quanta.

3.2 FB_N scheduling

The expression for the conditional mean waiting time for a fixed size quanta FB_N is derived in [21, 11]. In ([10], Chapter 4), different multilevel processor-sharing variants of FB_N are analyzed. In [11, 10] numerical evaluations for the conditional mean response time of jobs under FB_N are presented for the case of an M/M/1 system. We will derive the expressions for the conditional mean response time for the $M/G/1/FB_N$ model in this section.

In FB_N scheduling, a job in a queue j receives a service of one quantum (s_j) only if all queues i, $i < j$ are empty. Let τ_j be the service received by a job up to the foreground queue $j \leq N-1$, i.e., $\tau_j \triangleq \sum_{i=1}^{j} s_i$. A foreground job of size τ_j is delayed by the system workload due to all the jobs in the system ($W_o(\tau_j)$), truncated to a size τ_j if the job size is $x > \tau_j$. This workload is given by the Pollaczek-Khinchin (PK) mean value formula [10] applied to a job of size τ_j as:

$$W_o(\tau_j) = \frac{\lambda \overline{x^2_{\tau_j}}}{2(1-\rho_{\tau_j})}. \tag{3}$$

Note that $W_o \triangleq W_o(\infty)$ is the mean waiting time due to the total workload in the system. One can show that Equation (3) is the same as Equation (9) in [21], which is used to derive the expression for the conditional mean waiting time in FB_∞. Equation (3) is used in this paper because of its compact form. Putting Equation (3) in the expression of the mean waiting time of a foreground job of size τ_j as derived in [21, 11], we obtain the compact expression of $E[W(\tau_j)]$ as:

$$E[W(\tau_j)] = \frac{W_o(\tau_j) + \tau_{j-1}\rho_{\tau_{j-1}}}{(1-\rho_{\tau_{j-1}})}. \tag{4}$$

Since $E[T(\tau_j)] \triangleq E[W(\tau_j)] + \tau_j$, we obtain the expression for the conditional mean response time of a job of size τ_j under FB_N ($E[T(\tau_j|\tau_j \leq \tau_{N-1})]$) as:

$$\begin{aligned} E[T(\tau_j|\tau_j \leq \tau_{N-1})] &= \frac{W_o(\tau_j) - s_j\rho_{\tau_{j-1}}}{(1-\rho_{\tau_{j-1}})} \\ &+ \frac{\tau_j}{(1-\rho_{\tau_{j-1}})}. \end{aligned} \tag{5}$$

It is easy to see that as quantum sizes, $s_i\ \ i \in \{1, 2, ..., N\}$ approach to zero, Equation (5) becomes the same as the expression for $E[T(x)]_{FB_\infty}$ given in Equation (2).

Let us now look at background jobs. A background job returns at the head of the background queue $k_N \triangleq \lceil \frac{x-\tau_{N-1}}{s_N} \rceil$ times, and each time it is immediately taken back into service if all foreground queues are still empty. We denote the mean waiting time of a background job as $W_B(x)$, which is due to the total workload that the background job finds in the system upon its arrival (W_o) and the delay due to service interruptions caused by new arrivals while the background job is in the system ($W_s(\tau_{N-1})$). The time during which these new arrivals may occur has a mean duration of $E[T(x)] - s_N = W_B(x) + \tau_{N-1} + (k_N - 1)s_N$. Note that $E[T(x)] - s_N$ is also the duration during which a background job can be interrupted by new arrivals. The term $k_N - 1$ arises from the fact that a job is not interrupted once it begins its last quantum of service. The new arrivals have a mean arrival rate of λ. Hence, by Little's Law, the average number of these new arrivals is given as $\lambda(W_B(x) + [\tau_{N-1} + (k_N - 1)s_N]) = \lambda(E[T(x)] - s_N)$, each of which delays the background job by an average time of $\overline{x}_{\tau_{N-1}}$ time. Therefore, the expression of $W_s(\tau_{N-1})$ is given as $(W_B(x) + [\tau_{N-1} + (k_N - 1)s_N])\rho_{\tau_{N-1}}$, where $\rho_{\tau_{N-1}} = \lambda\overline{x}_{\tau_{N-1}}$. Hence,

$$W_B(x) = W_o + W_s(\tau_{N-1}), \tag{6}$$

substituting the expression of $W_s(\tau_{N-1})$ in Equation (6) we obtain:

$$W_B(x) = W_o + (W_B(x) + [\tau_{N-1} + (k_N - 1)s_N])\rho_{\tau_{N-1}},$$

after some algebra, we obtain the expression of $W_B(x)$ as:

$$W_B(x) = \frac{W_o + [\tau_{N-1} + (k_N - 1)s_N]\rho_{\tau_{N-1}}}{(1-\rho_{\tau_{N-1}})}. \tag{7}$$

We will use the expression of $W_s(\tau_{N-1})$ repeatedly in this paper, it is therefore convenient to provide its definition here as follows:

Definition 1 *Assume a background job with mean response time $E[T(x)]$, the delay of the job due to the service interruptions of new arrivals is defined as $W_s(\tau_{N-1}) \triangleq (E[T(x)] - s_N)\rho_{\tau_{N-1}}$.*

Using the relation that $E[T(x)] = E[W(x)] + x$, we obtain the expression for the conditional mean response time of a background job of size $x > \tau_{N-1}$ as:

$$\begin{aligned} E[T(x|x > \tau_{N-1})] &= \frac{W_o - s_N\rho_{\tau_{N-1}}}{(1-\rho_{\tau_{N-1}})} \\ &+ \frac{x}{(1-\rho_{\tau_{N-1}})}. \end{aligned} \tag{8}$$

Finally, we obtain the expression for the mean response time of a job under FB_N as:

$$E[T(x)] = \begin{cases} \text{Equation (5)} & \text{if } x \leq \tau_{N-1} \\ \text{Equation (8)} & \text{if } x > \tau_{N-1}. \end{cases} \tag{9}$$

Note from Equation (9) that the conditional mean response time of a job with size less than or equal to τ_{N-1} under FB_N is the same as its conditional mean response time under FB_∞. Therefore, a good tuning of τ_{N-1} value for a job size distribution with high CoV can guarantee that all small jobs receive the same service under FB_N as under FB_∞.

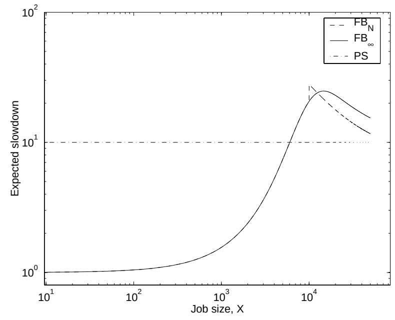
4 Numerical Evaluation of FB_N

In this section, we discuss numerical results of FB_N and we compare it with FB_∞ and PS for empirical job sizes with high and low CoV. Our objective is to evaluate FB_N in terms of reducing the response time of small jobs, and in terms of the amount of large jobs it penalizes. We use the bounded Pareto distribution $BP(k, p, \alpha)$ (where k and p are the minimum and maximum job sizes and α is the exponent of the power law) as a typical example of high CoV empirical job sizes for large p values and the exponential distribution to represent low CoV empirical job sizes. The density functions of the bounded Pareto and the exponential distributions are given as $f_{BP}(x) = \frac{\alpha k^{\alpha}}{1-(k/p)^{\alpha}} x^{-\alpha-1}$ for $k \leq x \leq p$ and $0 \leq \alpha \leq 2$, and $f_{Exp}(x) = \mu e^{-\mu x}$ for $x \geq 0$, $\mu \geq 0$ respectively.

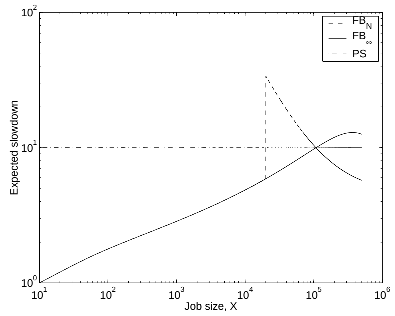
The BP distribution can have a very high CoV, whereas the CoV of the exponential distribution is always 1. In particular, we use the bounded Pareto $BP(10, 5 * 10^5, 1.1)$ with a mean of 72.7 and a $CoV = 19.8$ and the exponential distribution with a mean of $3 * 10^3$. $BP(10, 5 * 10^5, 1.1)$ distributed jobs have highly varying sizes as about 99% of jobs have small sizes and less than 1% of the largest jobs constitute about 50% of the total load. The number of foreground queues used in numerical analysis is $N - 1 = 2000$ and the quantum size is $s_i = 10, \forall i \in \{1, 2, ..., N\}$ for the BP distribution. Hence, the service required by a job that finishes service in the last foreground queue is $\tau_{N-1} = 20000$. We use the same quantum size values for the case of exponential distribution. The number of foreground queues used for the exponential distribution is $N - 1 = 1000$, which gives the service required by a job that finishes service in the last foreground queue as $\tau_{N-1} = 10000$.

Figures 1(a) and 1(b) show the mean slowdown of different job sizes at load $\rho = 0.9$. The slowdown as a function of job size in the figures exhibits three phases. The first phase when FB_N is identical to FB_∞, the second phase is when FB_N shows a sharp increase in slowdown and the conditional mean slowdown is higher under FB_N than under FB_∞, and the last phase is when FB_N performs better than FB_∞ in terms of their conditional mean slowdown. As noted in Equation (9), a foreground job under FB_N has the same response time (resp. slowdown) as under FB_∞. That is why, in the first phase of the figures, FB_N and FB_∞ have identical slowdown. However, we note that all foreground jobs have a lower mean slowdown under FB_∞ and FB_N than under PS for the BP distribution. For the exponential distribution, this is not the case.

We classify the background jobs to jobs that return to the background queue a few times and jobs that return to the background queue many times. The second phase represents the slowdown of the jobs that enter the background queue a few times. These jobs have a higher slowdown under FB_N than under FB_∞ or PS for both distributions, as they are no longer favored in the background queue. Instead, they are the smallest jobs in the background queue and thus are penalized by the FIFO policy in the background queue. The peak slowdown value of jobs in the second phase depends on the value of τ_{N-1} and on the load: the slowdown decreases for increasing



(a) *Exponential distribution*



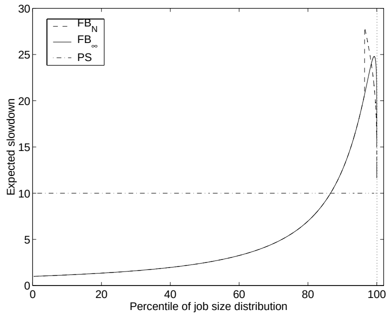
(b) $BP(10, 5 * 10^5, 1.1)$

Figure 1: *Expected conditional slowdown as a function of job size at load* $\rho = 0.9$

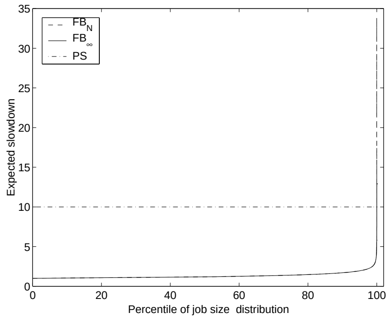
τ_{N-1} or decreasing load ρ values. If τ_{N-1} is small, the slowdowns (resp. response times) of the jobs in the background queue are affected by the service of a larger number of jobs in the background due to the FIFO scheduling order in the background queue. The dependence of $E[S(x)]$ on the load may be explained by the physical system behavior where the interference among jobs in the background queue is smaller when the arrival rate is small than when it is high.

For the jobs that return to the background queue many times, we observe from Figures 1(a) and 1(b) that FB_N results in a lower mean slowdown to the jobs than FB_∞. For BP distribution, we further see that these jobs have a lower mean slowdown under FB_N than under PS. This is not the case for the exponential distribution as Figure 1(a) shows that the mean slowdown of the jobs remains higher under FB_N than under PS. This is the third phase of the figures. This phenomenon is also observed to depend on the value of τ_{N-1} and on the load ρ. It is worth mentioning that the mean slowdown of these jobs under FB_N increases in increasing τ_{N-1} or load values. The numerical results showing the dependence of the mean slowdown on τ_{N-1} and on load are omitted here due to space limitation. We refer to [14] for more details.

We now analyze the percentage of large jobs that experience a higher mean slowdown under FB_N and FB_∞



(a) *Exponential distribution*

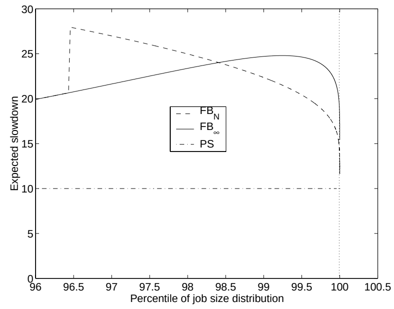


(b) $BP(10, 5*10^5, 1.1)$

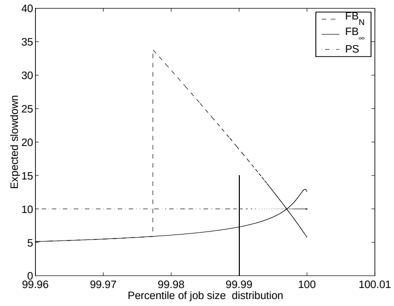
Figure 2: *Expected conditional slowdown* $E[S(x)]$ *as a function of percentiles of job size distribution, at load* $\rho = 0.9$

than under PS for the BP and exponential job size distributions. Figures 2(a) and 2(b) show the slowdown of FB_∞, FB_N, and PS as a function of the percentiles of job size distributions considered. Note that for the BP distribution, less that 1% of the largest jobs have a higher mean slowdown (but finite, see Figure 3(b)) under FB_∞ and FB_N than under PS. For the exponential distribution, we observe that about 15% of the jobs experience a higher slowdown under FB_∞ and FB_N than under PS. Hence, FB_∞ and FB_N are more fair for job size distributions with high CoVs than job size distributions with low CoVs.

For the BP distribution, we observe in Figure 3(b) that at load $\rho = 0.9$, less than 0.001% of the jobs receive a very slight penalty in terms of increase in their mean slowdown under FB_∞ as compared to PS, which is quite a small percentage. For the case of the exponential distribution (Figure 3(a)), the percentage of jobs that receive a penalty is observed to be higher than for the BP distribution. Finally, we observe in Figure 3(b) that FB_N is not fair for some jobs in the second phase with service times x slightly higher than τ_{N-1}. However, the percentage of these jobs is much lower (less than 0.02%) for the BP distribution that for exponential distribution and depends on the value of τ_{N-1}. Thus, for the job size distribution with



(a) *Exponential distribution*



(b) $BP(10, 5*10^5, 1.1)$

Figure 3: *Expected conditional slowdown* $E[S(x)]$ *as a function of percentiles of job size distribution zoomed at high percentiles, at load* $\rho = 0.9$

a high CoV, the results assert that FB_∞ and FB_N are quite fair.

5 FB_N under overload

The stability of FB_∞ under overload was proven in [15]. We showed that all jobs with sizes less than or equal to $x_{FB_\infty}(\lambda)$, with $\overline{x}_{x_{FB_\infty}(\lambda)} < \frac{1}{\lambda}$, receive service when FB_∞ system is overloaded. Since foreground jobs under FB_N have the same response times and slowdowns as under FB_∞, on average jobs in a foreground queue j can receive service under overload if $\overline{x}_{\tau_j} < \frac{1}{\lambda}$. On the other hand, jobs in the background queue have no chance of being serviced under overload even if all foreground jobs are serviced and $\overline{x}_{\tau_{N-1}} < \frac{1}{\lambda}$, since the FIFO policy applied to the background queue becomes unstable. The instability is due to the fact that when all foreground jobs receive service under overload condition, the load at the background queue $\rho - \lambda\overline{x}_{\tau_{N-1}}$ is always greater than the remaining effective load $1 - \lambda\overline{x}_{x_{FB_\infty}(\lambda)}$ hence the instability of the background queue. When the background queue is unstable, its size keeps growing to infinity. However, some jobs at the head of the queue can receive service and eventually leaves the system. Once again, this shows that the tuning of τ_{N-1} is important to make sure that small jobs receive service in the foreground queue so that they may also completely receive service under FB_N in case of overload. The tuning may require a prior knowledge of the job size distribution, which is difficult to obtain.

The numerical results for FB_N are summarized as follows :

- FB_N favors more jobs for job size distributions with high CoV than low CoV

- The percentage of large jobs that experience a penalty under FB_N is negligible for job size distributions with a high CoV, hence FB_N is quite fair to large jobs

- The maximum mean slowdown of the background jobs that return to the background queue a few times depends on the τ_{N-1} value and system load ρ: it increases in increasing load and decreasing τ_{N-1}

- Almost all background jobs will terminate receiving service under overload.

6 Service Differentiation in FB_N

FB_N can not differentiate jobs based on an attribute other than their size, and the jobs that enter the background queue a few times experience high response times under FB_N as load increases and even no service at all under overload. In many networking environments, service differentiation based on attributes such as protocol number, type of application, or user-assigned priorities is required to guarantee the quality of the important traffic. Examples of differentiation attributes are VPN traffic against IP public traffic, streaming traffic against elastic traffic, RTP against non-RTP, and etc. The service differentiation can also be based on more than one attribute. For example, we may want to give high priority to not only VPN traffic but also to delay intolerant streaming traffic. More service differentiation attributes are possible, and can be selected depending on which traffic the operator defines as more important. To achieve such a service differentiation, we propose variants of FB_N architecture that can classify the incoming jobs and differentiate their service.

These FB_N variants that we propose first classify the incoming jobs into *high priority* and *low priority* jobs. Then, the high priority jobs are favored over the low priority jobs. We denote the variables corresponding to high priority jobs by a subscript or superscript H and for the low priority jobs by a subscript or superscript L. The size of a high priority job is referred as x_H and that of a low priority job as x_L. We assume that a high priority job arrives at the system with probability p and a low priority job arrives with probability $1-p$. When λ is the average arrival rate of jobs in the system, the average arrival rates of high priority and low priority jobs are then $\lambda_H = p\lambda$ and $\lambda_L = (1-p)\lambda$ respectively. A reasonable mean arrival rate of the high priority jobs is at most 30% of the total mean arrival rate, i.e., $p \leq 0.3$. We further assume that the low and high priority jobs maintain the same distribution as the aggregate of the jobs. That is, if $f(x)$ is the p.d.f of all job classes, then $f(x) = f_L(x) = f_H(x)$. The load corresponding to high priority jobs of sizes less than or equal to x_H and low priority jobs of sizes less than or equal to x_L are $\rho_x^H = p\rho_x$ and $\rho_x^L = (1-p)\rho_x$ respectively. Similarly, we denote the system backlog due to high priority jobs as W_o^H and the backlog due to low priority jobs as W_o^L. The expressions for W_o^H and W_o^L are the same as Equation (3) with $\tau_{N-1} = \infty$, and with λ and ρ values corresponding to the class of the job. That is, $W_o^H = \frac{\lambda_H \overline{x_\infty^2}}{2(1-\rho_\infty^H)}$ and $W_o^L = \frac{\lambda_L \overline{x_\infty^2}}{2(1-\rho_\infty^L)}$.

In the next sections, we present two variants of FB_N. We derive the expressions for the conditional mean response times of foreground and background jobs with high priority and low priority. We also present some numerical results to compare the performance improvements in terms of reduction of the conditional mean response time for high priority jobs. We also evaluate the increase of the mean slowdown for the low priority jobs.

6.1 Fixed Priory FB_N Architecture

The first variant of FB_N that we propose is called *fixed priority* FB_N scheduling policy ($FP-FB_N$). $FP-FB_N$ employs a separate FB_N system for each priority class. In the $FP-FB_N$ policy, an incoming job is first classified to high priority or low priority, and then forwarded to the FB_N system that corresponds to his class. The jobs in each priority class are serviced in FB_N order except that the low priority jobs are serviced only if there are no high priority jobs, i.e., all queues in the FB_N that corresponds to high priority class are empty. Moreover, the low priority service is *preempted* on the arrival of the high priority jobs. Figure 4 shows the $FP-FB_N$ architecture. For simplicity in analysis, we assume that the number of queues, quantum values, and the values of τ_{N-1} in either FB_N of $FP-FB_N$ are the same.

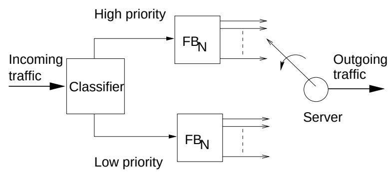


Figure 4: *Fixed priority FB_N architecture*

The $FP-FB_N$ scheduling policy improves the service of high priority jobs by avoiding the interruptions of the service of the high priority background jobs due to low priority jobs in the foreground queue and it reduces the mean response times of the low priority small jobs by servicing them in a separate FB_N policy. In the following sections, we compute the expressions of the conditional mean response times for jobs with different priorities.

6.1.1 High priority foreground jobs

The conditional mean response time of high priority foreground job under $FP-FB_N$ is the same as its conditional mean response time under FB_N with the mean arrival rate $\lambda = \lambda_H$ and load $\rho = \rho^H_\infty$. Hence, from Equation (5), we get the expression for conditional mean response time for a job size τ_j, $\forall j \in \{1, ..., N-1\}$ under $FP - FB_N$ ($E[T(x|\tau_j \leq \tau_{N-1})]_{FP-FB_N}$) as:

$$E[T(\tau_j|\tau_j \leq \tau_{N-1})] = \frac{W^H_o(\tau_j) - s_j \rho^H_{\tau_{j-1}}}{(1-\rho^H_{\tau_{j-1}})} + \frac{\tau_j}{(1-\rho^H_{\tau_{j-1}})}, \qquad (10)$$

where $W^H_o(\tau_j) = \frac{\lambda_H \overline{x^2_{\tau_j}}}{2(1-\rho^H_{\tau_j})}$, which is the system backlog due to the high priority jobs in the system that delays the high priority foreground job of size τ_j.

6.1.2 High priority background jobs

The response time of a high priority background job under $FP-FB_N$ is the same as the response time in an isolated FB_N policy with the mean arrival rate λ_H at load ρ^H_x. Hence, the expression for the mean response time of the job size x_H under $FP - FB_N$ is easily derived from Equation (8) as:

$$ET(x_H|x_H > \tau_{N-1}) = \frac{W^H_o - s_N \rho^H_{\tau_{N-1}}}{(1-\rho^H_{\tau_{N-1}})} + \frac{x_H}{(1-\rho^H_{\tau_{N-1}})}. \qquad (11)$$

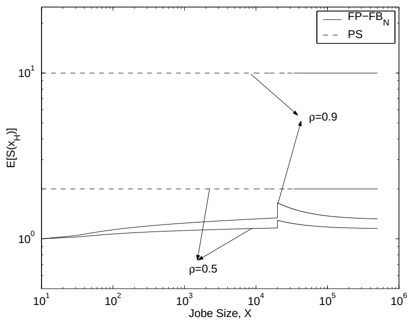


Figure 5: *Expected slowdown of high priority jobs under $FP - FB_N$ as a function of job size for $BP(10, 5*10^5, 1.1)$, at $p = 0.3$*

Figure 5 shows the mean slowdowns of the high priority jobs with mean arrival rate $\lambda_H = 0.3\lambda$ under $FP - FB_N$ at load $\rho = 0.9$ and $\rho = 0.5$. We see from the figure that $FP-FB_N$ significantly reduces the slowdown of the high priority jobs. We note that even at load $\rho = 0.9$, the mean slowdown of high priority jobs under $FP - FB_N$ is far below their mean slowdown under PS. Observe also that a reasonable mean arrival rate of the high priority jobs also guarantees that the high priority jobs will continue to receive service under overload. In particular, the high priority jobs receive service under overload as along as $\rho^H_\infty = p\rho < 1$.

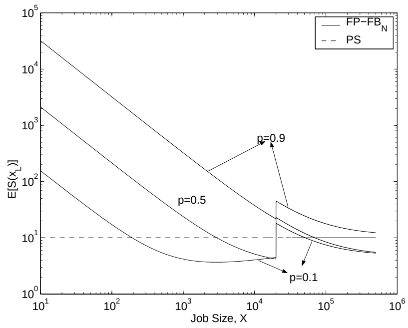


Figure 6: Expected slowdown of low priority jobs under $FP - FB_N$ as a function of job size for $BP(10, 5*10^5, 1.1)$, at load $\rho = 0.9$

6.1.3 Low priority foreground jobs

Assume an isolated low priority FB_N system. The mean waiting time of the low priority background job (x_L) in this isolated system is the same as its mean waiting time in an FB_N system with mean arrival rate and load of λ_L and ρ^L_x respectively. We denote this waiting time by $E[W(x_L)]$. In the $FP - FB_N$ policy however, the low priority foreground job will be further delayed by the service of the backlog that it finds in the high priority FB_N system upon its arrival W^H_o, the service of new arrivals of the high priority jobs while the low priority job is in the system $W_s(x_H)$, and its service time x_L. Hence,

$$\begin{aligned} E[T(x_L|x_L \leq \tau_{N-1})] &= W^H_o + E[W(x_L)] \\ &+ (W_s(x_H) + x_L). \end{aligned}$$

The expressions for W^H_o and $E[W(x_L)]$ are given in Equation (3) for $\tau_j = \infty$ and $\rho_{\tau_j} = \rho^H_\infty$ and Equation (4) for $\lambda = \lambda_L$ and $\rho_{\tau_{j-1}} = \rho^L_{\tau_{j-1}}$ respectively. Similarly, $W_s(x_H)$ is given by Definition 1 as $W_s(x_H) = (E[T(x_L|x_L \leq \tau_{N-1})] - s_N)\rho^H_\infty$). Then,

$$\begin{aligned} E[T(x_L|x_L \leq \tau_{N-1})] &= W^H_o + E[W(x_L)] + x_L \\ &+ E[T(x_L|x_L \leq \tau_{N-1})]\rho^H_\infty \\ &- s_N \rho^H_\infty, \end{aligned}$$

after some algebra, we obtain $E[T(x_L|\tau_{N-1} \geq x_L)]$ as:

$$E[T(x_L|x_L \leq \tau_{N-1})] = \frac{W^H_o + E[W(x_L)]}{(1-\rho^H_\infty)} + \frac{x_L - s_N \rho^H_\infty}{(1-\rho^H_\infty)}.$$

6.1.4 Low priority background jobs

Finally, a low priority background job is delayed by the system backlog due to the high priority jobs that it finds in the system upon arrival, W_o^H, the service of new arrivals of high priority jobs $W_s(x_H)$, and its service time x_L. In addition, the job will wait in the system due to its waiting time in a low priority FB_N system assuming that it is isolated $W_B(x_L)$. That is,

$$\begin{aligned} E[T(x_L|x_L > \tau_{N-1})] &= W_o^H + W_B(x_L) \\ &+ W_s(x_H) + x_L. \end{aligned}$$

The expression for $W_B(x_L)$ is given in Equation (7) for $W_o = W_o^L$ and $\rho = \rho_{\tau_{N-1}}^L$ and $W_s(x_H)$ is given by Definition 1 as $(E[T(x_L|x_L > \tau_{N-1})] - s_N)\rho_\infty^H$. Hence, the expression for $E[T(x_L|x_L > \tau_{N-1})]$ is given as:

$$\begin{aligned} E[T(x_L|x_L > \tau_{N-1})] &= W_o^H + W_B(x_L) \\ &+ E[T(x_L|x_L > \tau_{N-1})]\rho_\infty^H \\ &- (s_N\rho_\infty^H - x_L), \end{aligned}$$

after some algebra, we obtain:

$$\begin{aligned} E[T(x_L|x_L > \tau_{N-1})] &= \frac{W_o^H + W_B(x_L)}{(1-\rho_\infty^H)} \\ &+ \frac{x_L - s_N\rho_\infty^H}{(1-\rho_\infty^H)}. \qquad (12) \end{aligned}$$

Figure 6 shows the mean slowdown of the low priority jobs under $FP - FB_N$ for different p values at load $\rho = 0.9$. We observe from the figure that the mean slowdown of the low priority small jobs under $FP - FB_N$ is quite high and increases in increasing the mean arrival rate of high priority jobs ($\lambda_H = p\lambda$). Note that the low priority jobs experience the minimum mean response time under $FP - FB_N$ when $p = 1$, i.e., there are only low priority jobs in the system. The minimum mean response time of low priority jobs under $FP - FB_N$ is the same as under FB_N. Thus, the service of the high priority jobs under $FP - FB_N$ comes at the expense of a high penalty to small jobs with low priority.

6.2 Differential FB_N

Figures 2(b) and 3(b) show that if the job size distribution exhibits high variability, more than 99% of jobs have a lower slowdown under FB_N than under PS. Thus, without differentiation, all small jobs under the FB_N policy receive low mean response times. It is only a few background jobs that receive high response times, particularly at load values close to 1. In this section, we propose and analyze another variant of FB_N that we call *Differential* FB_N ($DF - FB_N$). The objective of this policy is to improve the mean response time of high priority background jobs while maintaining the response time of all small jobs as low as under FB_N. $DF - FB_N$ services all small jobs in foreground queues, but low priority and high priority background jobs are serviced in separate background queues. Hence, $DF - FB_N$ maintains two background queues, see Figure 7. Low priority background jobs are serviced in a low priority background queue only if all foreground queues and the high priority background queue are empty, whereas the high priority background jobs x_L^H receive service once all foreground queues are empty. In the next section, we analyze $DF - FB_N$ assuming that the mean arrival rate of the high priority jobs λ_H and the mean arrival rate of low priority jobs λ_L are known.

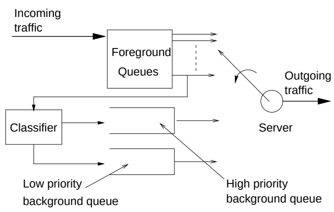


Figure 7: *Differential* FB_N *architecture*

6.2.1 Foreground jobs

The conditional mean response time of a foreground job (high priority or low priority) under $DF - FB_N$ is the same as the conditional mean response time under FB_N with the same mean arrival rate. Hence, the formula for the mean response time of a foreground job that completes service in queue j, $j \in \{1, ..., N-1\}$ ($E[T(\tau_j|\tau_j \leq \tau_{N-1})]$) is the same as Equation (5) with appropriate mean arrival rate λ and load ρ.

6.2.2 High priority background jobs

Now, we compute the expression for the mean response time of high priority background jobs ($E[T(x_H|x_H > \tau_{N-1})]$). A high priority background job is delayed in the queue due to the service of the system backlog of the high priority jobs that it finds in the system W_o^H, the service of the backlog of low priority jobs in the foreground queues $W_o^L(\tau_{N-1})$, and its own service x_H. In addition, the job is delayed by the service of the newly arriving jobs in foreground queues by $W_s(\tau_{N-1})$. That is,

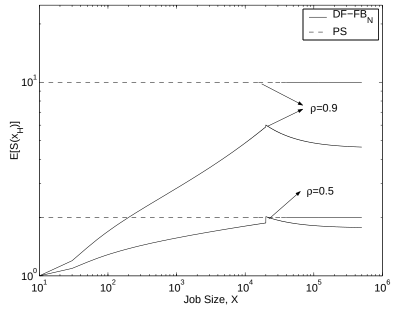
$$\begin{aligned} E[T(x_H|x_H > \tau_{N-1})] &= W_o^H + W_o^L(\tau_{N-1}) \\ &+ W_s(\tau_{N-1}) + x_H. \end{aligned}$$

Definition 1 gives the expression of $W_s(\tau_{N-1})$ as $(E[T(x_H|x_H > \tau_{N-1})] - s_N)\rho_{\tau_{N-1}}$ and the expression for $W_o^L(\tau_{N-1})$ is given from Equation (3) for $\tau_j = \tau_{N-1}$ and $\lambda = \lambda_L$. Hence, $E[T(x_H|x_H > \tau_{N-1})]$ is given as:

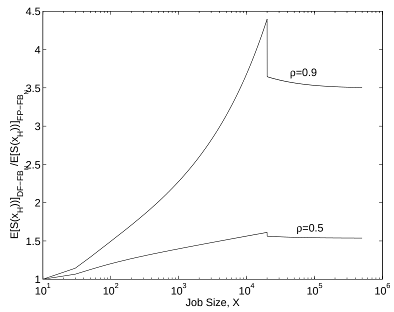
$$\begin{aligned} E[T(x_H|x_H > \tau_{N-1})] &= W_o^H + W_o^L(\tau_{N-1}) + \\ &\quad E[T(x_H|x_H > \tau_{N-1})]\rho_{\tau_{N-1}} \\ &\quad - s_N\rho_{\tau_{N-1}} + x_H, \end{aligned}$$

after some algebra, we get:

$$E[T(x_H|x_H > \tau_{N-1})] = \frac{W_o^H + W_o^L(\tau_{N-1})}{(1-\rho_{\tau_{N-1}})} + \frac{x_H - s_N\rho_{\tau_{N-1}}}{(1-\rho_{\tau_{N-1}})}.$$



(a) $E[S(x_H)]$



(b) $\frac{E[S(x_H)]_{DF-FB_N}}{E[S(x_H)]_{FP-FB_N}}$

Figure 8: *The performance of $DF-FB_N$ for $BP(10, 5*10^5, 1.1)$ as a function of job size, at $p = 0.3$*

In the $DF-FB_N$ architecture, the service of all new arriving jobs in foreground queues interrupt the service of high priority background jobs. However, when the job size distribution has a high variance, the load constituted by these small jobs is small, less than half of the total load. Hence, differentiating the service of only background jobs has a positive impact on the the mean response time of high priority background jobs. Figure 8(a) compares the mean slowdown of high priority jobs $E[S(x_H)]$ under $DF-FB_N$ and PS for the BP distribution at load $\rho = 0.5$ and $\rho = 0.9$, and $p = 0.3$. We see that for both considered load values, $DF-FB_N$ offers a lower mean slowdown than PS for all high priority jobs. Figure 8(b) shows the ratio of the mean slowdown of high priority jobs under $DF-FB_N$ to their mean slowdown of the jobs under $FP-FB_N$. We observe that $\frac{E[S(x_H)]_{DF-FB_N}}{E[S(x_H)]_{FP-FB_N}}$ reaches as high as above 4 at load $\rho = 0.9$, which means that the maximum conditional mean slowdown under $DF-FB_N$ is 4 times higher than the conditional mean slowdown under $FP-FB_N$. The maximum value of the ratio is quite low, about 1.5 for load $\rho = 0.5$. The performance difference between $DF-FB_N$ and $FP-FB_N$ in terms of reducing the mean response time of high priority jobs is not very significant. And this is accounted by the fact that $FP-FB_N$ maintains as double as the number of queues as $DF-FB_N$.

6.2.3 Low priority background jobs

Finally, the mean response time of a low priority background job $E[T(x_L|\tau_{N-1} < x_L)]$ is a result of its waiting time due to the backlog that it finds in the system upon its arrival W_o , its service time x_L, and the average waiting time due to service interruptions of newly arriving jobs. The service of these newly arriving jobs that affect the response time of the low priority background job are the service of the low priority jobs in the foreground queues $W_s^L(\tau_{N-1})$ and the service of all new arrivals of high priority jobs $W_s^H(\infty)$. That is,

$$E[T(x_L|\tau_{N-1} < x_L)] = W_o + W_s^H(\infty) + W_s^L(\tau_{N-1}) + x_L.$$

Definition 1 gives the expressions for $W_s^H(\infty)$ and $W_s^L(\tau_{N-1})$ as $(E[T(x_L|x_L > \tau_{N-1})] - s_N)\rho_\infty^H$ and $(E[T(x_L|x_L > \tau_{N-1})] - s_N)\rho_{\tau_{N-1}}^L$ respectively. Hence,

$$\begin{aligned} E[T(x_L|\tau_{N-1} < x_L)] &= W_o + x_L \\ &+ E[T(x_L|x_L > \tau_{N-1})]\rho_\infty^H \\ &+ E[T(x_L|x_L > \tau_{N-1})]\rho_{\tau_{N-1}}^L \\ &- (s_N\rho_\infty^H + s_N\rho_{\tau_{N-1}}^L), \end{aligned}$$

simplifying the above equation, we get

$$E[T(x_L|x_L > \tau_{N-1})] = \frac{W_o - s_N(\rho_\infty^H + \rho_{\tau_{N-1}}^L)}{(1-\rho_\infty^H - \rho_{\tau_{N-1}}^L)} + \frac{x_L}{(1-\rho_\infty^H - \rho_{\tau_{N-1}}^L)}.$$

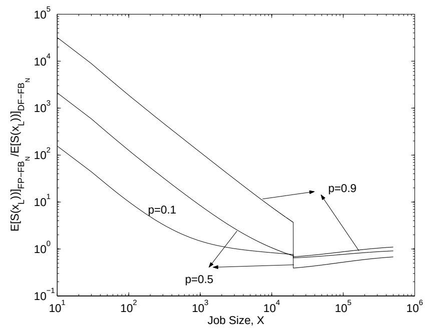


Figure 9: $\frac{E[S(x_L)]_{FP-FB_N}}{E[S(x_L)]_{DF-FB_N}}$ for $BP(10, 5*10^5, 1.1)$ as a function of job size, at load $\rho = 0.9$

We now look at the improvement of $DF-FB_N$ over $FP-FB_N$ in terms of reducing the mean response time of low priority jobs for the BP distribution considered. This is shown in Figure 9 where we plot the ratio of the mean slowdown between the policies. We see that, while there is no big difference between the mean slowdown of low priority background jobs offered by both policies, $DF - FB_N$ significantly reduces the mean slowdown of the jobs. Moreover, under overload, high priority background jobs under $DF - FB_N$ receive service as long as the load due to the service received in foreground queues is less than 1, which is the case by our definition of highly varying job sizes. Therefore, we conclude that $DF - FB_N$ is a more suitable policy than a $FP - FB_N$ with respect to improving the mean response times of all jobs in the system.

7 Conclusion

This paper demonstrates the impact of the variability of job sizes on the performance of a size-based scheduling policy called foreground background with N queues (FB_N), and modifies FB_N to differentiate services based on desired attributes in addition to the job sizes. We show through analysis that the mean response time of jobs under $M/G/1/FB_N$ system is significantly reduced and the percentage of jobs that are penalized under the system is negligibly low when job sizes have a high *coefficient of variability* ($CoV > 1$) compared to job sizes with a low $CoV = 1$. However, FB_N cannot differentiate the services of jobs based on attributes other than their sizes and its ability to service jobs under overload is limited to foreground queues only. Therefore, FB_N can't guarantee low response time for important jobs or users (particularly with large job sizes).

We propose and analyze two variants of FB_N that classify the incoming jobs, before servicing them, into high and low priority based on any desired attributes. These policies are referred to as *fixed priority* FB_N ($FP - FB_N$) and *differential* FB_N ($DF - FB_N$). Numerical results conducted for empirical job sizes with a high CoV show that $FP - FB_N$ offers absolute guarantees to the response time of high priority jobs at the expense of a high penalty for low priority small jobs. Similarly, $DF - FB_N$ also guarantees the service of the high priority jobs. In addition, $DF - FB_N$ maintains the mean response time of the low priority foreground jobs as low as the mean response time under FB_N. In contrast to FB_N, both $DF - FB_N$ and $FP - FB_N$ can also guarantee the service of the high priority jobs under overload at any reasonable mean arrival rate of the high priority jobs.

We observe that the size-based scheduling analyzed in this paper have the potential to reduce delay in the network while offering service differentiation when the traffic objects (jobs) have highly varying service times.

References

[1] N. Bansal and M. Harchol-Balter, "Analysis of SRPT Scheduling: Investigating Unfairness", In *Sigmetrics 2001 / Performance 2001*, pp. 279–290, june 2001.

[2] P. Barford and M. E. Crovella, "Generating Representative Web Workloads for Network and Server Performance Evaluation", In *Proceedings of Performance'98/SIGMETRIS'98*, pp. 151–169, July 1998.

[3] M. Bender et al., "Flow and Stretch Metrics for Scheduling Continuous Job Streams", In *Proceedings of the 9th Annual ACM-SIAM Symposium on Discrete Algorithms*, 1998.

[4] E. G. Coffman and P. J. Denning, *Operating Systems Theory*, Prentice-Hall Inc., 1973.

[5] M. Crovella and A. Bestavros, "Self-Similarity in World Wide Web Traffic: Evidence and Possible Causes", *IEEE/ACM Transactions on Networking*, pp. 835–846, December 1997.

[6] M. E. Crovella, "Performance Evaluation with Heavy Tailed Distributions", In *Job Scheduling Strategies for Parallel Processing 2001 (JSSPP)*, pp. 1–10, 2001.

[7] M. Harchol-Balter and A. Downey, "Exploiting Process Lifetime Distributions for Dynamic Load Balancing", *ACM Transactions on Computer Systems*, 15(3):253–285, 1997.

[8] M. Harchol-Balter et al., "Connection Scheduling in Web Servers", In *USENIX Symposium on Internet Technologies and Systems (USITS '99)*, pp. 243–254, October 1999.

[9] M. Harchol-Balter, "The Effect of Heavy-Tailed Job Size. Distributions on Computer System Design", In *Proc. of ASA-IMS Conf. on Applications of Heavy Tailed Distributions in Economics*, June 1999.

[10] L. Kleinrock, *Queuing Systems, Volume II: Computer Applications*, Wiley, New York, 1976.

[11] H. Krayl, E. J. Neuhold, and C. Unger, *Grundlagen der Betriebssysteme*, Walter de Gruyter, Berlin, New York, 1975.

[12] W. E. Leland and T. J. Ott, "Load-balancing Heuristics and Process Behavior", In *Proceedings of Performance and ACM Sigmentrics*, pp. 54–69, May 1986.

[13] V. Paxson, "Emperically-derived Analytic Models of Wide-Area TCP Connections", *IEEE/ACM Transactions on Networking*, 2(4):316–336, August 1994.

[14] I. A. Rai, G. Urvoy-Keller, and E. W. Biersack, "Comparison Study of PS, SET, and Foreground Background (FB_N) Scheduling Policies", TR 01.11.1, Institut Eurecom, November 2001.

[15] I. A. Rai, G. Urvoy-Keller, and E. W. Biersack, "FB_∞: An Efficient Scheduling Policy for Edge Routers to Speedup the Internet Access", TR 02.04.1, Institut Eurecom, April 2002, Submitted for Publication.

[16] I. A. Rai, G. Urvoy-Keller, and E. W. Biersack, "On Reducing Response Time for VPN Traffic", TR 02.03.1, Institut Eurecom, March 2002, Submitted for Publication.

[17] J. Roberts and L. Massoulie, "Bandwidth Sharing and Admission Control for Elastic Traffic", In *ITC Specialist Seminar*, 1998.

[18] L. E. Schrage, "The queue M/G/1 with feedback to lower priority queues", *Management Science*, 13(7):466–474, 1967.

[19] L. E. Schrage and L. W. Miller, "The queue M/G/1 with the shortest processing remaining time discipline", *Operations Research*, 14:670–684, 1966.

[20] A. Shaikh, J. Rexford, and K. G. Shin, "Load-sensitive Routing of Long-lived IP Flows", In *Proc. ACM SIGCOMM*, pp. 215–226, September 1999.

[21] R. W. Wolff, "Time Sharing with Priorities", *SIAM Journal of Applied Mathematics*, 19(3):566–574, 1970.