Classifying Scheduling Policies with respect to Unfairness in an M/GI/1

Adam Wierman [1]   Mor Harchol-Balter [2]

October 2002

CMU-CS-02-187

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Abstract

It is common to classify scheduling policies based on their mean response times. Another important, but sometimes opposing, performance metric is a scheduling policy's fairness. For example, a policy that biases towards short jobs so as to minimize mean response time, may end up being unfair to long jobs. In this paper we define three types of unfairness and demonstrate large classes of scheduling policies that fall into each type. We end with a discussion on which jobs are the ones being treated unfairly.

[1]Carnegie Mellon University, Computer Science Department. Email: acw@cs.cmu.edu.
[2]Carnegie Mellon University, Computer Science Department. Email: harchol@cs.cmu.edu.
This work was supported by NSF Career Grant CCR-0133077 and by Pittsburgh Digital Greenhouse Grant 01-1.

Keywords: scheduling; unfairness; M/G/1; FB; LAS; SET; feedback; least attained service; shortest elapsed time; PS; processor sharing; SRPT; shortest remaining processing time; slowdown.

1 Introduction

Traditionally the performance of scheduling policies has been measured using mean response time (i.e. sojourn time, time in system) [9, 10, 12, 15], and more recently mean slowdown [1, 4, 5]. Under these measures, size based policies that give priority to small job sizes at the expense of larger sizes perform quite well [13]. However, these policies tend not to be used in practice due to a fear of unfairness. For example, a policy that always biases towards jobs with small sizes is likely to treat jobs with large sizes unfairly [3, 16, 17, 18].

This tradeoff between minimizing mean response time while maintaining fairness is an important design constraint in many applications. For example, in the case of Web servers, it has been shown that by giving priority to requests for small files, a Web server can significantly reduce response times; however it is important that this improvement not come at the cost of unfairness to requests for large files [6, 9]. The same tradeoff applies to other application areas; for example, scheduling in supercomputing centers. Here too it is desirable to get short jobs out quickly, while not penalizing the large jobs, which are typically associated with the important customers. The tradeoff also occurs for age based policies. For example, UNIX processes are assigned decreasing priority based on their current age – CPU usage so far. This can create unfairness for old processes. To address the tension between minimizing mean response time and maintaining fairness, hybrid scheduling policies have also been proposed; for example, policies that primarily bias towards young jobs, but give sufficiently old jobs high priority as well.

Recently, the topic of unfairness has been looked at formally by Bansal and Harchol-Balter, who study the unfairness properties of the Shortest-Remaining-Processing-Time (SRPT) policy under an M/GI/1 system [2]; and by Harchol-Balter, Sigman, and Wierman, who address unfairness under all scheduling policies asymptotically as the job size grows to infinity [7]. In this paper, these results are extended to characterize the existence of unfairness under all priority based scheduling policies, for all job sizes.

In order to begin to understand unfairness however, we must first formalize what is meant by fair performance. In this definition, and throughout this paper we will be using the following notation. We will consider only an M/GI/1 system with a continuous service distribution having finite mean and finite variance. We let $T(x)$ be the steady-state response time for a job of size x, and $\rho < 1$ be the system load. That is $\rho \stackrel{\text{def}}{=} \lambda E[X]$, where λ is the arrival rate of the system and X is a random variable distributed according to the service distribution $F(x)$ with density function $f(x)$. The slowdown seen by a job of size x is $S(x) \stackrel{\text{def}}{=} T(x)/x$, and the expected slowdown for a job of size x under scheduling policy P is $E[S(x)]^P$.

Definition 1.1 *Jobs of size x are treated fairly under policy P iff $E[S(x)]^P = 1/(1-\rho)$. Further, a scheduling policy is fair iff it treats every job size fairly.*

Definition 1.2 *Jobs of size x are treated unfairly under policy P iff $E[S(x)]^P > 1/(1-\rho)$. Further, a scheduling policy is unfair iff there exists a job size x that is treated unfairly.*

This definition is a natural extension of the notion of fairness used in [2, 7]. Notice that this definition of fairness has two parts. First, the expected slowdown seen by a job of size x must be a constant (i.e.

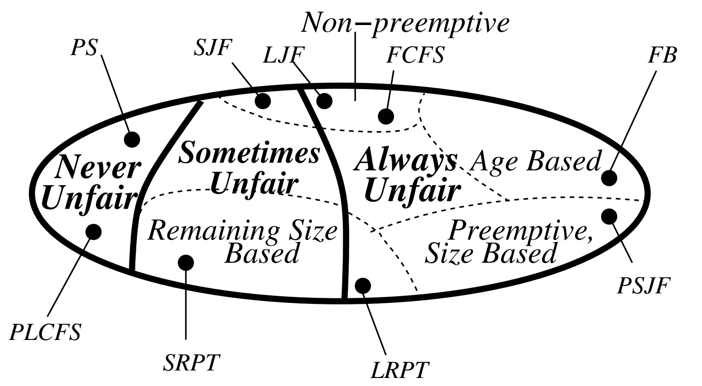

Figure 1: *Classification of unfairness showing a few examples of both individual policies and groups of policies within each class.*

independent of x). Processor-Sharing (PS) is a common scheduling policy that achieves this. Under PS the processor is shared evenly among all jobs in the system at every point in time. It is well known that $E[S(x)]^{PS} = 1/(1-\rho)$ [20], independent of the job size x. The second condition of the definition of fairness is that the particular constant must be $1/(1-\rho)$. Although this constant may seem arbitrary, in Section 2 we will show that $1/(1-\rho)$ is the lowest possible constant obtainable under any policy with constant expected slowdown. This fact is a formal verification that $1/(1-\rho)$ is the appropriate constant for defining fairness.

With these definitions, it is now possible to classify scheduling policies based on whether they (i) treat all job sizes fairly or (ii) treat some job sizes unfairly. Curiously, we find that some policies may fall into either type (i) or type (ii) depending on the system load. We therefore define *three classes of unfairness*:

Never Unfair: Policies under which, for all loads, no job size is treated unfairly.

Sometimes Unfair: Policies under which, for some loads, some job size is treated unfairly; but under which for other loads, no job size is treated unfairly.

Always Unfair: Policies under which, for all loads, there is some job size that is treated unfairly.

The goal of this paper is to classify scheduling policies into the above three types (see Figure 1). Scheduling policies are typically divided into non-preemptive policies and preemptive policies. We find that non-preemptive policies can either be Sometimes Unfair or Always Unfair, however preemptive policies may fall into any of the three types. In this paper, we concentrate on preemptive priority based policies. These include policies for which (i) a fixed priority is associated with each possible job size (a.k.a. *size based policies*), (ii) a fixed priority is associated with each possible job age (a.k.a. *age based policies*), and (iii) a fixed priority is associated with each possible remaining size (a.k.a. *remaining size based policies*). Observe that (i) includes policies like Preemptive-Shortest-Job-First where short jobs have higher priority, but also includes perverse policies like Preemptive-Longest-Job-First and others. Observe that (ii) includes policies like Feedback scheduling where young jobs are given priority, yet also includes other practical policies that primarily bias towards young jobs and also give high priority to sufficiently old jobs. Observe that (iii) includes policies like Shortest-Remaining-Processing-Time-First and Longest-Remaining-Processing-Time-First that bias towards jobs with short and long remaining time respectively, as well as practical hybrids. We show that all policies in (i) and (ii) are Always Unfair; whereas policies in (iii) can be Sometimes Unfair or Always Unfair.

Lastly, for the case where jobs are being treated unfairly, we investigate *which job sizes* are treated unfairly, and find that these are not necessarily the jobs one would expect. Furthermore, we find that the answer depends on the system load.

2 Never Unfair

Two well known Never Unfair policies are Processor-Sharing (`PS`) and Preemptive-Last-Come-First-Served (`PLCFS`). Recall that `PLCFS` always devotes the full processor to the most recent arrival. Both of these policies have the same expected performance: $E[S(x)]^{PS} = E[S(x)]^{PLCFS} = 1/(1-\rho)$ for all x. An important open problem not answered in this paper is the question of what other policies are in the Never Unfair class.

We now address why the value of $1/(1-\rho)$ appears in the definition of Never Unfair. It seems possible that there is a policy that is both *fair* in the sense that all job sizes have the same expected slowdown, and has slowdown strictly less than $1/(1-\rho)$. We show below that there is no such policy.

Theorem 2.1 *There is no policy P such that $E[S(x)]^P$ is independent of x and $E[S(x)]^P < 1/(1-\rho)$.*

This theorem follows from the lemma below, which provides a necessary condition for a policy to be Never Unfair. This necessary condition will be appealed to in the proof of Theorem 4.1.

Lemma 2.1 *If scheduling policy P is Never Unfair, then $\lim_{x\to\infty} E[S(x)]^P = 1/(1-\rho)$*

Proof : First, notice that because P is Never Unfair, $\lim_{x\to\infty} E[S(x)]^P \leq 1/(1-\rho)$. Thus, we need only show that $\lim_{x\to\infty} E[S(x)]^P \geq 1/(1-\rho)$. We accomplish this by bounding the expected slowdown for a job of size x from below, and then showing that the lower bound converges to $1/(1-\rho)$ as we let $x \to \infty$.

To lower bound the expected slowdown, we consider a modified policy $Q_{x,a}$ that throws away arrivals both of size greater than x and of size $y < x$ such that $T(y) \geq a$. Further, $Q_{x,a}$ works on the remaining jobs at the exact moments P works on these jobs. We will begin by calculating the load made up of jobs of size less than y in this system, $\rho(y)^{Q_{x,a}}$. We can use Markov's Inequality to obtain a probabilistic bound that the response time for a job of size y is less than some value a: $P(T(y) < a) \geq 1 - \frac{y}{a(1-\rho)}$. Thus, we see that

$$\rho(y)^{Q_{x,a}} = \lambda \int_0^y \left(1 - \frac{t}{a(1-\rho)}\right) tf(t)dt = \rho(y) + \frac{\lambda m_2(y)}{a(1-\rho)}$$

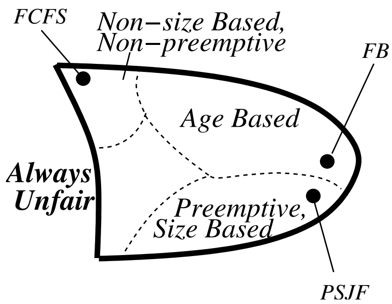

Figure 2: *A detail of the Always Unfair classification.*

where $\rho(y) \stackrel{\text{def}}{=} \lambda \int_0^y tf(t)dt$ is the load made up by jobs of size less than or equal to y in P and $m_2(y) \stackrel{\text{def}}{=} \int_0^y t^2 f(t)dt$. The key observation of this proof is that as a, y, and x get very large, $\rho(y)^{Q_{x,a}}$ approaches ρ.

As a lower bound on the response time for a job of size x under P, we can consider that up until the moment x has remaining time a, x is allowed to work whenever $Q_{x,a}$ is idle of jobs of size less than y. This occurs whenever the system is empty of jobs of size less than or equal to some size y that will finish within a time. At this point, x is allowed to complete uninterrupted. This provides a lower bound on the response time of P because all work that we are accounting for must be done under P in order for P to be Never Unfair.

$$\begin{aligned} E[T(x)]^P &\geq \frac{x-a}{1-\rho(y)^{Q_{x,a}}} + a \\ E[S(x)]^P &\geq \frac{x-a}{x\left(1-\rho(y)+\frac{\lambda m_2(y)}{a(1-\rho)}\right)} + \frac{a}{x} \end{aligned}$$

Now, we must set y and a as functions of x such that, as we let $x \to \infty$, we converge as desired. Notice that as $x \to \infty$, we would like $\rho(y) \to \rho$, $\frac{\lambda m_2(y)}{a(1-\rho)} \to 0$, and $\frac{a}{x} \to 0$. Thus, we must have $a \ll x$ and $y \ll x$ such that $y \to \infty$ and $a \to \infty$. We can accomplish this by setting $a = 4\sqrt{x}$ and $y = \sqrt{x}$. Notice that $m_2(\sqrt{x}) \to E[X^2] < \infty$ as $x \to \infty$. Now, looking at expected slowdown we see that as $x \to \infty$:

$$E[S(x)]^P \geq \frac{x-4\sqrt{x}}{x\left(1-\rho(\sqrt{x})+\frac{\lambda m_2(\sqrt{x})}{4\sqrt{x}(1-\rho)}\right)} + \frac{4\sqrt{x}}{x} \to \frac{1}{1-\rho}$$

■

3 Always Unfair

In this section we will show that a large number of common policies are Always Unfair. That is, many common policies are guaranteed to treat some job size unfairly under all system loads. In each subsection we will investigate a class of common policies, proving that the class is Always Unfair. Figure 2 summarizes the policies that will be looked at in this section.

Section 3.1 illustrates that all non-preemptive policies are Always Unfair when the service distribution is defined on some neighborhood of zero. However, if the service distribution has a non-zero lower bound then only non-size based, non-preemptive policies are guaranteed to be Always Unfair. Section 3.2 shows that any preemptive, size based policy is Always Unfair. In fact, we show that any job size that is given a fixed, low initial priority upon arrival will be treated unfairly. We next discuss policies where a job's priority is a function of its current age. We first investigate a common policy of this type in Section 3.3 and then in Section 3.4 extend the results to show that every age based policy is Always Unfair.

3.1 Non-size based, non-preemptive policies

The analysis in this section is based on the simple observation that any policy where a small job cannot preempt the job in service will likely be unfair to small jobs. For example, let us begin with the class of non-preemptive policies.

Lemma 3.1 *Any non-preemptive policy P is Always Unfair under any service distribution defined on a neighborhood of zero.*

Proof : We can bound the performance of P by noticing that, at a minimum, an arriving job of size x must take x time plus the excess of the job that is serving. Thus, $E[T(x)]^P \geq x + \frac{\rho E[X^2]}{2E[X]}$. Notice that $\lim_{x\to 0} E[S(x)]^P = \infty$. Thus, there exists some y such that $E[S(y)]^P > 1/(1-\rho)$, for all $\rho < 1$. ∎

This proof can be generalized quite a bit. For any policy P such that at least $q > 0$ fraction of the time a small job x does not preempt a large job in service, $E[S(x)]^P$ will have a term dependent on $E[X^2]$ which will cause $E[S(x)] \to \infty$ as $x \to 0$. Such policies are Always Unfair when the service distribution is defined on a neighborhood of zero. These include all policies where some fraction of the arriving jobs are tagged as high priority, others are tagged as low priority, and low priority jobs cannot preempt high priority jobs. Under such differential service policies, those small jobs of low priority will be treated unfairly if the service distribution is defined on a neighborhood around zero.

However, under general service distributions a much smaller set of policies can be classified as Always Unfair. These are the non-size based, non-preemptive policies. (Note that the remainder of the possible non-preemptive policies are explored in Section 4.1.)

Theorem 3.1 *All non-size based, non-preemptive policies P are Always Unfair.*

Proof : Assume that the service time distribution has lower bound $C > 0$ (we have already dealt with the case of $C = 0$). We will show that jobs of size C are treated unfairly. Recall that all such policies have the same expected response time [8].

$$\begin{aligned} E[T(C)]^P &= C + \frac{\lambda E[X^2]}{2(1-\rho)} = \frac{C(1-\rho) + \lambda \int_0^\infty (t+C)\overline{F}(t+C)dt}{1-\rho} \\ &= \frac{C - C\rho + C\rho + \lambda \int_0^\infty t\overline{F}(t+C)dt}{1-\rho} > \frac{C}{1-\rho} \end{aligned}$$

where the last inequality follows since the service distribution is required to be non-deterministic. ∎

3.2 Preemptive, size based policies

In this section we analyze size based policies (i.e. policies where a job receives a priority based on its original size), where higher priority jobs always preempt lower priority jobs. Note one such policy, Preemptive-Shortest-Job-First (`PSJF`), improves overall time in system with respect to `PS` by biasing towards jobs with small sizes. It is important to understand the unfairness properties caused by this bias. Further, every policy in this class will bias against a particular job size, so it is important to understand if unfairness results from this bias.

Theorem 3.2 *Any preemptive, size based policy is Always Unfair.*

The remainder of this section will prove this result. We will break the analysis into two cases: (1) when there exists a finite sized job that has the lowest priority and (2) when there is no finite sized job with the lowest priority. Case (2) will be broken into two subcases: (2.1) when priorities decrease monotonically (i.e., the `PSJF` policy), and (2.2) when priorities are non-monotonic, but have no finite sized job that receives the lowest priority. This method of proof will be used again in Section 3.4 and Section 4.3.

It will be helpful in the proofs below if we first analyze the Longest-Remaining-Processing-Time (`LRPT`) policy. At any given point, the `LRPT` policy shares the processor evenly among all the jobs in the system with the longest remaining processing time. `LRPT` has the following expected slowdown [7]:

$$E[S(x)]^{LRPT} \quad = \quad \frac{1}{1-\rho} + \frac{\lambda E[X^2]}{2x(1-\rho)^2} = \frac{E[B(x)]}{x} + \frac{E[B(V)]}{x}$$

where V is the work in the system seen by an arrival and $B(x)$ is the length of a busy period started by a job of size x. Recall that $E[V] = \frac{\lambda E[X^2]}{2(1-\rho)}$ under all work conserving policies.

Lemma 3.2 *Under LRPT, for all finite job sizes y, $E[S(y)]^{LRPT} > 1/(1-\rho)$ under any bounded or unbounded service distribution, for all ρ. Further, $E[S(y)]^{LRPT}$ is monotonically decreasing with y to $1/(1-\rho)$.*

We are now ready to prove case (1).

Lemma 3.3 *Any preemptive, size based policy P that gives some finite job size y the lowest possible priority is Always Unfair.*

Proof : We will derive the time a job of size y spends in the system. Let $T(y) = W(y) + R(y)$ where $W(y)$ is the time until y first receives service (waiting time) and $R(y)$ is the time from when y first receives service until it completes (residence time). Notice that y must wait behind all jobs that are already in the system. So, its waiting time is $W(y) = B(V)$. Further, since an arriving job will preempt the job w.p.1, we know that the residence time $R(y) = B(y)$.

Thus, for jobs of the lowest priority $E[S(y)]^P = E[S(y)]^{LRPT}$. Because `LRPT` has a monotonically decreasing expected slowdown curve that converges to $1/(1-\rho)$, we can conclude that no matter what job size has the lowest priority, the expected slowdown of that job size will be strictly greater than $1/(1-\rho)$. ■

We now move to case (2.1).

Lemma 3.4 *Under PSJF there is some job size y such that for all $x > y$ and for all ρ, $E[S(x)]^{PSJF} > 1/(1-\rho)$ under any unbounded service distribution.*

Proof : It is well known that [8]:

$$E[T(x)]^{PSJF} = \frac{\lambda \int_0^x t^2 f(t)dt}{2(1-\rho(x))^2} + \frac{x}{1-\rho(x)} \text{ where } \rho(x) \stackrel{\text{def}}{=} \lambda \int_0^x t f(t)dt.$$

Thus, $\lim_{x\to\infty} E[S(x)]^{PSJF} = 1/(1-\rho)$ since the service distribution is assumed to have finite variance. To prove the lemma it is sufficient to show that $\frac{d}{dx}E[S(x)]$ converges to zero from below as $x \to \infty$.

By observing that

$$\frac{d}{dx}E[S(x)]^{PSJF} = \frac{d}{dx}\frac{E[T(x)]^{PSJF}}{x} = \frac{x\frac{d}{dx}E[T(x)]^{PSJF} - E[T(x)]^{PSJF}}{x^2}$$

our goal reduces to showing that as $x \to \infty$

$$x\frac{d}{dx}E[T(x)]^{PSJF} - E[T(x)]^{PSJF} < 0 \tag{1}$$

Let us begin by calculating

$$x\frac{d}{dx}E[T(x)]^{PSJF} = \frac{\lambda^2 x^2 f(x)\int_0^x t^2 f(t)dt}{2(1-\rho(x))^3} + \frac{3\lambda x^3 f(x)}{2(1-\rho(x))^2} + \frac{x}{1-\rho(x)}$$

which gives us

$$x\frac{d}{dx}E[T(x)]^{PSJF} - E[T(x)]^{PSJF} = \frac{\lambda^2 x^2 f(x)\int_0^x t^2 f(t)dt}{2(1-\rho(x))^3} + \left(\frac{3\lambda x^3 f(x)}{2(1-\rho(x))^2} - \frac{\lambda\int_0^x t^2 f(t)dt}{2(1-\rho(x))^2}\right)$$

Observe that distributions with finite second moments must have $f(x) = o(x^{-3})$, where $g(x) \stackrel{\text{def}}{=} o(h(x))$ if $\lim_{x\to\infty}\frac{g(x)}{h(x)} = 0$. Using this observation, we see that

$$\lim_{x\to\infty} x\frac{d}{dx}E[T(x)]^{PSJF} - E[T(x)]^{PSJF} = 0 + \left(0 - \frac{\lambda E[X^2]}{2(1-\rho)^2}\right) < 0$$

Recalling Equation 1, we can conclude that $E[S(x)] \to 1/(1-\rho)$ from above as $x \to \infty$. ■

We are now left with only case (2.2).

Lemma 3.5 *Any preemptive, size based policy P where there is no finite job size that receives the smallest priority is Always Unfair.*

Proof : Note that Lemma 3.4 leaves only the case where for every job size x there is a job size $y > x$ such that the priority of y is less than the priority of x, but the priorities are not decreasing monotonically.

We will complete the proof by taking advantage of our knowledge of PSJF. Choose some job size y such that PSJF treats all job sizes larger than y unfairly. We know that for some size z greater than y, z has a lower priority than all jobs of smaller size. Thus, z is treated, with respect to these smaller jobs, as if it were in PSJF. Further, if jobs larger than z have higher priority than z, they will simply raise $E[S(z)]^P$. Thus, z is treated at least as badly as it would have been under PSJF. Since any such z is treated unfairly under PSJF (by Lemma 3.3), this completes the proof. ■

Notice that under the policies in this section, the job sizes that are treated unfairly depend on how priorities are assigned. When there is a finite job size y that receives the lowest priority, then y is treated unfairly. However, in the case when no job size was given the lowest priority, we see that it is not the largest job that is treated the most unfairly. This follows from the fact that $\frac{d}{dx}E[S(x)]^{PSJF}$ is decreasing as $x \to \infty$. Thus, some other class of large, but not the largest, jobs is receiving the most unfair treatment. This observation is discussed in more detail in Section 3.3.2.

3.3 FB

We now turn to a specific policy, Feedback (FB) scheduling. Under FB, the job with the least attained service gets the processor to itself. If several jobs all have the least attained service, they time-share the processor via PS. This is a practical policy, since a job's age is always known, although its size may not be known. This policy improves upon PS with respect to mean response time and mean slowdown when the job size distribution has decreasing failure rate [19]. We have [8]:

$$E[T(x)]^{FB} = \frac{\lambda \int_0^x t\overline{F}(t)dt}{(1-\rho_x)^2} + \frac{x}{1-\rho_x} \text{ where } \rho_x \overset{\text{def}}{=} \lambda \int_0^x \overline{F}(t)dt.$$

Given the bias that FB provides for small jobs (they are always young), it is natural to ask about the performance of the large jobs. Thus, understanding the growth of slowdown as a function of the job size x is important. The following Lemma will be useful in evaluating FB's performance.

Lemma 3.6 *For all x and ρ, $E[T(x)]^{PSJF} \leq E[T(x)]^{FB}$.*

Proof : The proof is simply algebraic.

$$\begin{aligned} E[T(x)]^{PSJF} &= \frac{\lambda \int_0^x t^2 f(t)dt}{2(1-\rho(x))^2} + \frac{x}{1-\rho(x)} \\ &\leq \frac{\lambda E[X_x^2]}{2(1-\rho(x))^2} + \frac{x}{1-\rho(x)} \\ &\leq \frac{\frac{1}{2}\lambda E[{X_x}^2] + x(1-\rho_x)}{(1-\rho_x)^2} = E[T(x)]^{FB} \end{aligned}$$

■

Theorem 3.3 *Under FB scheduling there is some job size y such that for all $x > y$, $E[S(x)]^{FB} > 1/(1-\rho)$ under any service distribution, for all ρ. Furthermore, $E[S(x)]^{FB}$ is not monotonic in x.*

Proof : The first part of the theorem follows immediately from combining Lemma 3.4 and Lemma 3.6.

For the second part, we show that $E[S(x)]^{FB}$ is monotonically increasing for small x, but decreasing as $x \to \infty$. We start by differentiating response time:

$$x\frac{d}{dx}E[T(x)]^{FB} = \frac{2\lambda^2\overline{F}(x)x\int_0^x t\overline{F}(t)dt}{(1-\rho_x)^3} + \frac{2\lambda x^2\overline{F}(x)}{(1-\rho_x)^2} + \frac{x}{1-\rho_x}$$

which gives us

$$x\frac{d}{dx}E[T(x)]^{FB} - E[T(x)]^{FB} = \left(\frac{2\lambda^2\overline{F}(x)x\int_0^x t\overline{F}(t)dt}{(1-\rho_x)^3}\right) + \left(\frac{2\lambda x^2\overline{F}(x)}{(1-\rho_x)^2} - \frac{\lambda\int_0^x t\overline{F}(t)dt}{(1-\rho_x)^2}\right) \quad (2)$$

Recall from Equation 1 that the above gives us the sign of $\frac{d}{dx}E[S(x)]^{FB}$.

There are two terms in Equation 2. The first term is clearly positive. Notice that for x such that $\overline{F}(x) \geq \frac{1}{4}$ we have:

$$x\frac{d}{dx}E[T(x)]^{FB} - E[T(x)]^{FB} \geq \frac{\lambda}{(1-\rho_x)^2}\left(2x^2\overline{F}(x) - \frac{1}{2}x^2\right) \geq 0 \quad (3)$$

which shows that $E[S(x)]^{FB}$ is monotonically increasing for x such that $F(x) \leq \frac{3}{4}$.

We now prove that the expected slowdown converges to $1/(1-\rho)$ from above as $x \to \infty$. First, we know that $\lim_{x\to\infty} E[S(x)]^{FB} = 1/(1-\rho)$ [7]. Next, Equation 2 gives us the sign of $\frac{d}{dx}E[S(x)]^{FB}$. As in the proof of Lemma 3.4, for any distribution with finite second moment, we know that $\overline{F}(x) = o(x^{-2})$. Using this observation,

$$\lim_{x\to\infty} x\frac{d}{dx}E[T(x)]^{FB} - E[T(x)]^{FB} = -\frac{\lambda E[X^2]}{2(1-\rho)^2} < 0$$

Thus, there exists some job size x_0 such that for all $x > x_0$, $E[S(x)]^{FB}$ is monotonically decreasing in x. ■

The proof of this theorem shows us that all job sizes greater than a certain size are have higher mean response time under `FB` than under `PS`. Counter-intuitively however, the job that performs the worst is not the largest job. Thus, the intuition that by helping the small jobs `FB` must hurt the biggest jobs is not entirely true.

Interestingly, this theorem is counter to the common portrayal of `FB` in the literature. When investigating $E[S(x)]^{FB}$, previous literature has used percentile plots such as Figure 3(b), which hide the behavior of the largest one percent of the jobs [11]. When we look at the same plots as a function of job size, such as Figure 3(a), the presence of a hump becomes evident. In fact, even under bounded distributions, this hump exists, seemingly regardless of the bound placed on x.

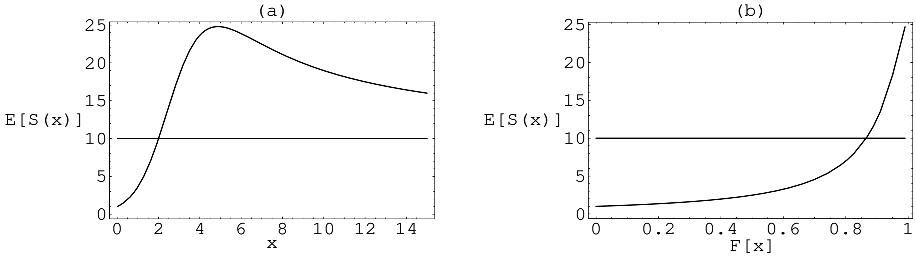

Figure 3: *Plots (a) and (b) show the growth of* $E[S(x)]^{FB}$ *for* $\rho = .9$. *In both cases the service distribution is taken to be Exponential with mean 1. The horizontal line shows fair performance, thus when* $E[S(x)]^{FB}$ *is above this line FB is treating a job size unfairly.*

3.3.1 Who is treated unfairly?

Having shown that some job sizes are treated unfairly under FB scheduling, it is next interesting to understand exactly which job sizes are seeing poor performance. The following theorem places a lower bound on the size of jobs that can be treated unfairly.

Theorem 3.4 *For* x *such that* $\rho_x \leq 1 - \sqrt{1-\rho}$, $E[T(x)]^{FB} \leq 1/(1-\rho)$.

Proof : The proof will proceed by simply manipulating $E[T(x)]^{FB}$.

$$\begin{aligned} E[T(x)]^{FB} &= \frac{\lambda \int_0^x t\overline{F}(t)dt}{(1-\rho_x)^2} + \frac{x}{1-\rho_x} \leq \frac{\lambda x \int_0^x \overline{F}(t)dt}{(1-\rho_x)^2} + \frac{x}{1-\rho_x} \\ &= \frac{\rho_x x}{(1-\rho_x)^2} + \frac{x(1-\rho_x)}{(1-\rho_x)^2} = \frac{x}{(1-\rho_x)^2} \end{aligned}$$

Letting $\rho_x \leq 1 - \sqrt{1-\rho}$ we complete the proof of the theorem. ■

It is important to notice that as ρ increases, so does the lower bound $1 - \sqrt{1-\rho}$ on ρ_x. In fact, this bound converges to 1 as $\rho \to 1$, which signifies that the size of the smallest job that might be treated unfairly is increasing unboundedly as ρ increases. Interestingly, this work also provides bounds on the job sizes that might be treated unfairly under PSJF due to Lemma 3.6.

3.3.2 Intuition for non-monotonicity

The fact that FB and PSJF have non-monotonic slowdown is somewhat surprising. Below we provide an intuitive explanation for this phenomenon.

For small jobs, it is clear that FB and PSJF provide preferential treatment. Thus it is believable that the slowdown should increase monotonically as job size increases.

Next consider a somewhat large job x, of size x, where this job is large enough that with high probability it is the largest job in any busy period in which it appears. Under FB and PSJF, job x will complete only at the end of the busy period, since it is the largest job in the busy period. Observe that job x will also only

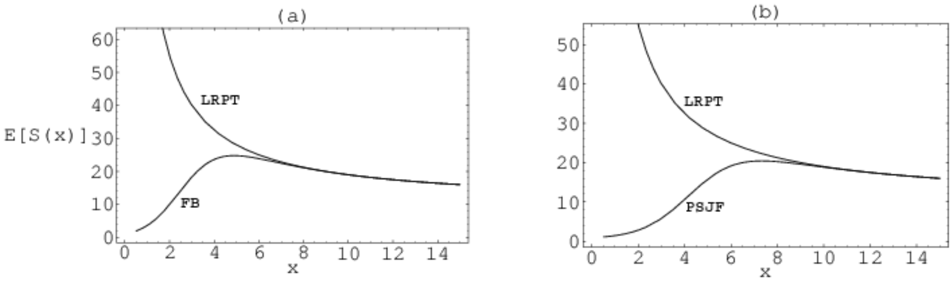

Figure 4: *Plot (a) shows $E[S(x)]^{LRPT}$ (above) and $E[S(x)]^{FB}$ (below). Plot (b) shows $E[S(x)]^{LRPT}$ (above) and $E[S(x)]^{PSJF}$ (below). In both cases $\rho = .9$ and the service distribution is taken to be Exponential with mean 1. Notice that the expected slowdown for a job of size x under both FB and PSJF quickly converges to the expected slowdown of x under LRPT.*

complete at the end of its busy period under LRPT, since all jobs complete at the end of the busy period under LRPT. Thus the performance of job x under FB and PSJF may be approximated by the performance of job x under LRPT. Next recall from Lemma 3.2, that the expected slowdown of job x under LRPT converges monotonically from above to $1/(1-\rho)$ as $x \to \infty$. Thus it is plausible that the expected slowdown of job x under FB and PSJF also converges monotonically from above to $1/(1-\rho)$ as $x \to \infty$.

Figure 4(a) shows that FB does in fact converge in performance to LRPT for large job sizes. Figure 4(b) shows the same for PSJF.

3.4 Age based policies

FB scheduling is one example of an age based policy (i.e. policies where a job's priority is some function of its current age). Age based policies are interesting because they include many hybrid policies where, in order to minimize mean response time and curb the unfairness seen by large jobs, both sufficiently old jobs and very young jobs receive preferential treatment.

Note that, as in FB, we will choose to break ties among jobs in the system with the same priority according to FCFS. Thus, when multiple jobs have the same age, and priority is increasing with age, the job that arrived first will be worked on alone for some period of time; however when multiple jobs have the same priority and priority is decreasing with age, this leads to PS among the jobs with the same priority.

Theorem 3.5 *Age based policies are Always Unfair.*

The remainder of this section will prove this theorem using a method similar to the method used in Section 3.2. We break the analysis into two cases: (1) the case when there exists a finite sized job that has the lowest priority and (2) when there is no finite sized job with the lowest priority. We begin with case (1).

Lemma 3.7 *Any age based policy P where there is a finite age C that receives the lowest priority is Always Unfair.*

Proof : We will show that P must be unfair to a job of size C^+, where C^+ is infinitesimally larger than C.

First notice that when a job of size C^+ arrives, all the work in the system can be guaranteed to be completed before C^+ leaves. Further, all arriving jobs of size x will have $\min\{x, C\}$ work completed on them before C^+ leaves the system. Thus:

$$E[T(C^+)]^P = \frac{\frac{\lambda E[X^2]}{2(1-\rho)} + C^+}{1-\rho_C} = \frac{\lambda E[X^2]}{2(1-\rho)(1-\rho_C)} + \frac{C^+}{1-\rho_C}$$

Now, notice that $E[T(C^+)]^P > C^+/(1-\rho)$ when

$$\frac{\lambda}{2}E[X^2] > C^+ (\rho - \rho_C)$$

or equivalently

$$(1-\rho) + \frac{\lambda E[X^2]}{2C^+} > 1 - \rho_C$$

Since $(1-\rho) \geq (1-\rho(C))$, the above condition is met for all finite C. ■

We now move to case (2).

Lemma 3.8 *Any age based policy where no finite job size has the lowest priority is Always Unfair.*

The proof of this final lemma follows from Theorem 3.3 and an argument symmetric to the proof of Lemma 3.5.

4 Sometimes Unfair

We now move to the class of Sometimes Unfair policies – policies that for some ρ treat all job sizes fairly, but for other ρ treat some job size unfairly. In Section 4.1 we return to non-preemptive policies and illustrate that when the service distribution sets a non-zero lower bound on the smallest job size, non-preemptive policies can avoid being Always Unfair, but cannot attain Never Unfair. In Section 4.2 we show that under the Shortest-Remaining-Processing-Time (SRPT) policy for $\rho \leq \frac{1}{2}$, $E[S(x)]^{SRPT}$ is monotonically increasing in x for all x and is always $\leq 1/(1-\rho)$. However, for $\rho > \rho_{crit}$, $E[S(x)]^{SRPT}$ is monotonically increasing in x for all x such that $\rho(x) \leq \frac{1}{2}$ and is monotonically decreasing in x for all x greater than some x_0. We also contrast the behavior of SRPT under bounded versus unbounded service distributions. More generally, in Section 4.3 we show that any remaining size based policy is either Sometimes Unfair or Always Unfair.

4.1 Non-preemptive, size-Based Policies

This section completes the analysis of non-preemptive policies begun in Section 3.1. It is based on the observation that if there is a lower bound on the smallest job size in the service distribution, then it is possible for a non-preemptive policy to avoid being Always Unfair.

Theorem 4.1 *Any non-preemptive, size-based policy P is either Sometimes Unfair or Always Unfair.*

Proof : Recall that $\lim_{x\to\infty} E[S(x)]^Q = 1$ for all non-preemptive policies Q, by Theorem 4 from [7]. Thus, we can apply Lemma 2.1 to conclude that a non-preemptive policy Q cannot attain Never Unfair. Thus, P (being a non-preemptive policy) must be either Always Unfair or Sometimes Unfair.

Observe there are examples of size based, non-preemptive policies in each of the two classes. For instance, it can easily be shown that the Longest-Job-First (LJF) policy is Always Unfair. However, Shortest-Job-First (SJF) is only Sometimes Unfair – that is, there exist service distributions and loads such that $E[S(x)]^{SJF} \le 1/(1-\rho)$ for all x. One example of such a distribution and load is $(X-2) \sim Exp(1)$ with $\rho = 0.2$. ■

4.2 SRPT

Under the SRPT policy, at every moment of time, the server is processing the job with the shortest remaining processing time. The SRPT policy is well-known to be optimal for minimizing mean response time [14]. The mean response time for a job of size x is as follows [13]:

$$\begin{aligned} E[T(x)]^{SRPT} &= \frac{\frac{\lambda}{2}\int_0^x t^2 f(t)dt + \frac{\lambda}{2}x^2\overline{F}(x)}{(1-\rho(x))^2} + \int_0^x \frac{dt}{1-\rho(t)} \\ &= \frac{\lambda\int_0^x t\overline{F}(t)dt}{(1-\rho(x))^2} + \int_0^x \frac{dt}{1-\rho(t)} \end{aligned}$$

where $\rho(x) \stackrel{\text{def}}{=} \lambda\int_0^x tf(t)dt$.

Theorem 4.2 *For x such that $\rho(x) \le \frac{1}{2}$, $E[S(x)]^{SRPT}$ is monotonically increasing in x.*

Proof : Begin by defining

$$m_2(x) \stackrel{\text{def}}{=} \int_0^x t^2 f(t) = 2\int_0^x t\overline{F}(t)dt - 2x^2\overline{F}(x)$$

Then we can derive

$$x \cdot \frac{d}{dx}E[T(x)]^{SRPT} = \frac{2\lambda^2 f(x)x^2\int_0^x t\overline{F}(t)dt}{(1-\rho(x))^3} + \frac{\lambda x^2\overline{F}(x)}{(1-\rho(x))^2} + \frac{x}{1-\rho(x)}$$

which gives us

$$\begin{aligned} x \cdot \frac{d}{dx}E[T(x)]^{SRPT} - E[T(x)]^{SRPT} &= \left(\frac{2\lambda^2 f(x)x^2\int_0^x t\overline{F}(t)dt}{(1-\rho(x))^3}\right) + \left(\frac{\lambda x^2\overline{F}(x)}{(1-\rho(x))^2} - \frac{\lambda\int_0^x t\overline{F}(t)dt}{(1-\rho(x))^2}\right) \\ &\quad + \left(\frac{x}{1-\rho(x)} - \int_0^x \frac{dt}{1-\rho(t)}\right) \\ &= \left(\frac{2\lambda^2 f(x)x^2\int_0^x t\overline{F}(t)dt}{(1-\rho(x))^3}\right) - \left(\frac{\lambda m_2(x)}{2(1-\rho(x))^2}\right) + \left(\frac{x}{1-\rho(x)} - \int_0^x \frac{dt}{1-\rho(t)}\right) \end{aligned}$$

Recall that this expression provides us with the sign of the derivative of slowdown. There are 3 terms in the above expression. The first of these terms is clearly positive. The third of these terms is also clearly positive. We will complete the proof by showing that the third term is of larger magnitude than the second term.

To obtain a bound on the third term, we can quickly show that

$$\begin{aligned} \frac{x}{1-\rho(x)} - \int_0^x \frac{dt}{1-\rho(t)} &= \int_0^x \frac{(1-\rho(t)) - (1-\rho(x))}{(1-\rho(t))(1-\rho(x))} dt \\ &\geq \frac{1}{1-\rho(x)} \int_0^x \rho(x) - \rho(t) dt \end{aligned} \tag{4}$$

To further specify this bound we can compute

$$\begin{aligned} \int_0^x \rho(t)dt &= \lambda \int_0^x \int_0^t sf(s)dsdt = \lambda \int_0^x \int_s^x sf(s)dtds \\ &= \lambda \int_0^x sf(s)(x-s)ds = \rho(x)x - \lambda m_2(x) \end{aligned} \tag{5}$$

Finally, putting all three terms back together we see that when $\rho(x) \leq \frac{1}{2}$,

$$\begin{aligned} x \cdot \frac{d}{dx} E[T(x)]^{SRPT} - E[T(x)]^{SRPT} &= \left(\frac{2\lambda^2 f(x) x^2 \int_0^x t\overline{F}(t)dt}{(1-\rho(x))^3} \right) - \left(\frac{\lambda m_2(x)}{2(1-\rho(x))^2} \right) + \left(\frac{x}{1-\rho(x)} - \int_0^x \frac{dt}{1-\rho(t)} \right) \\ &\geq -\left(\frac{\lambda m_2(x)}{2(1-\rho(x))^2} \right) + \left(\frac{\lambda m_2(x)}{1-\rho(x)} \right) \geq 0 \end{aligned}$$

■

Corollary 4.1 *If $\rho \leq \frac{1}{2}$, $E[S(x)]^{SRPT}$ is monotonically increasing for all x. Furthermore $E[S(x)]^{SRPT} \leq 1/(1-\rho)$ for all x.*

Proof : This follows immediately from the above theorem and by recalling the following result: for any work conserving scheduling policy P, $\lim_{x\to\infty} E[S(x)]^P \leq 1/(1-\rho)$ [7]. ■

The observation above that $E[S(x)]^{SRPT} \leq 1/(1-\rho)$ for all x when $\rho < \frac{1}{2}$ was proven in [2] using a different technique that did not describe the behavior of $E[S(x)]^{SRPT}$ as a function of increasing x.

The previous theorem showed monotonically increasing slowdown for SRPT under low load. We now show that if load is sufficiently high, the opposite behavior occurs.

Theorem 4.3 *There exists a $\rho_{crit} < 1$ such that for all $\rho > \rho_{crit}$, $E[S(x)]^{SRPT}$ has monotonically decreasing slowdown for $x \geq x_o$, for some x_o. Further, for $\rho > \rho_{crit}$, for all $x > x_o$, $E[S(x)]^{SRPT} > 1/(1-\rho)$ under any unbounded service distribution.*

Earlier work (see Theorem 8 of [2]) showed that for a *bounded job size distribution*, the largest job size p has the property that $E[S(p)]^{SRPT} > 1/(1-\rho)$. The above theorem extends this result to unbounded

job size distributions by utilizing monotonicity. The monotonicity result above is somewhat surprising. One might assume that the largest jobs are the ones receiving the most unfair treatment under `SRPT`. This is in fact the case for *bounded* job size distributions, however it is not true for *unbounded* job size distributions.
Proof :

The proof for the unbounded case is somewhat technical, but will follow a similar method to the previous proof. We will show that as $x \to \infty$ the derivative of expected slowdown approaches zero from below.

As in Equation 1, the main section of the proof will again look at $x \cdot \frac{d}{dx} E[T(x)]^{SRPT} - E[T(x)]^{SRPT}$. To evaluate the above expression, we need to evaluate Equation 4. Because evaluating the integral in this expression is difficult, we apply the Mean Value Theorem, which tells us that there exists a $c_x \in [0, x]$ such that

$$\begin{aligned} \frac{1}{1-\rho(x)} \int_0^x \frac{\rho(x)-\rho(t)}{1-\rho(t)} dt &= \frac{1}{(1-\rho(x))(1-\rho(c_x))} \int_0^x \rho(x)-\rho(t) dt \\ &= \frac{\lambda m_2(x)}{(1-\rho(x))(1-\rho(c_x))}. \end{aligned}$$

Thus we have:

$$\begin{aligned} x \cdot \frac{d}{dx} E[T(x)]^{SRPT} - E[T(x)]^{SRPT} &= \left(\frac{2\lambda^2 f(x) x^2 \int_0^x t\overline{F}(t)dt}{(1-\rho(x))^3} \right) - \left(\frac{\frac{\lambda}{2} m_2(x)}{(1-\rho(x))^2} \right) + \left(\frac{x}{1-\rho(x)} - \int_0^x \frac{dt}{1-\rho(t)} \right) \\ &\to -\frac{\frac{\lambda}{2} E[X^2]}{(1-\rho)^2} + \frac{\lambda E[X^2]}{(1-\rho)(1-\rho(c_\infty))} \quad \text{as } x \to \infty \end{aligned}$$

So, the derivative of slowdown converges from below when this is less than zero, which occurs when

$$\begin{aligned} 1-\rho(c_\infty) &> 2-2\rho \\ \text{or equivalently, } \rho &> \frac{1+\rho(c_\infty)}{2} \end{aligned}$$

To complete the proof, we need to bound $\rho(c_\infty)$. By showing that $\rho(c_\infty) < 1$ we illustrate a ρ_{crit} such that when $\rho > \rho_{crit}$, $E[S(x)]^{SRPT}$ will not be monotonic in x.

To understand what $\rho(c_x)$ is we let $x > 0$ and notice

$$\begin{aligned} \int_0^x \rho(x)-\rho(t)dt &\leq \int_0^x \frac{\rho(x)-\rho(t)}{1-\rho(t)} dt \leq \frac{1}{1-\rho(x)} \int_0^x \rho(x)-\rho(t)dt \\ 1 &\leq \frac{\int_0^x \frac{\rho(x)-\rho(t)}{1-\rho(t)} dt}{\int_0^x \rho(x)-\rho(t)dt} \leq \frac{1}{1-\rho(x)} \end{aligned}$$

So, c_x is such that

$$\begin{aligned} \frac{1}{1-\rho(c_x)} &= \frac{\int_0^x \frac{\rho(x)-\rho(t)}{1-\rho(t)} dt}{\int_0^x \rho(x)-\rho(t)dt} \\ \rho(c_x) &= 1 - \frac{\int_0^x \rho(x)-\rho(t)dt}{\int_0^x \frac{\rho(x)-\rho(t)}{1-\rho(t)} dt} \end{aligned}$$

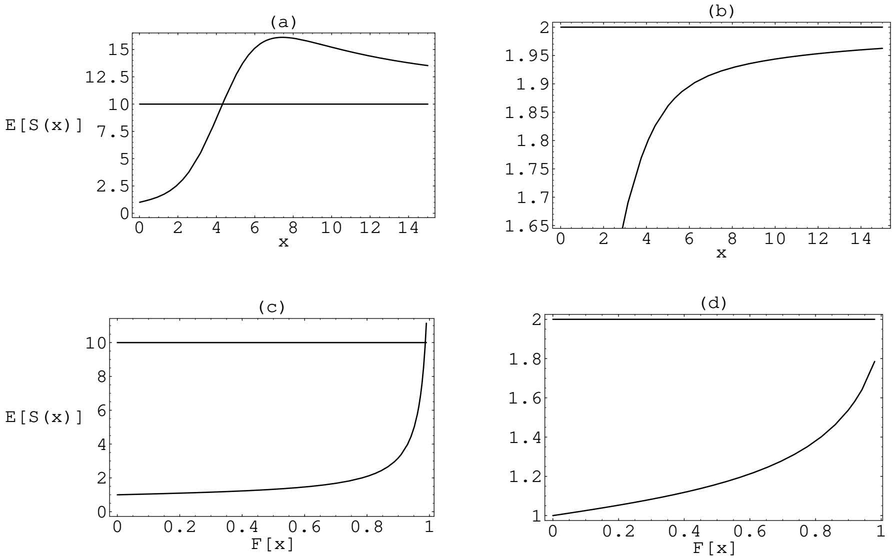

Figure 5: *Plots (a) and (c) show the growth of* $E[S(x)]^{SRPT}$ *for* $\rho = .9$*, while (b) and (d) show* $E[S(x)]^{SRPT}$ *when* $\rho = .5$*. In both cases the service distribution is taken to be Exponential with mean 1. The horizontal line shows fair performance, thus when* $E[S(x)]^{SRPT}$ *is above this line SRPT is treating a job size unfairly.*

Thus there exists a $\rho_{crit} < 1$ when the second term is bounded away from 1. The remainder of the proof bounds this value away from 1. Because the remainder of the proof is algebraic, we leave it in Appendix A. ■

The existence of this x_0 size beyond which $E[S(x)]^{SRPT}$ is monotonically decreasing has gone unnoticed by previous research. The reason is that percentile plots are typically used when viewing expected slowdown. As seen in Figure 5, because the hump occurs around the 99th percentile it is hidden when looking at the percentile plots. Viewing those same plots as a function of job size, such as in Figure 5 (a) and (b), reveals the existence of a hump under high load. Note that the peak of the hump occurs far from the largest job size.

4.2.1 Who is treated unfairly?

Having seen that SRPT is Sometimes Unfair, it is interesting to consider which job sizes are being treated fairly/unfairly. The following theorem shows that as ρ increases, the number of jobs being treated fairly also increases.

Theorem 4.4 *For* x *such that* $\rho(x) \le \max\{1-\sqrt{1-\rho}, \frac{1}{2}\}$, $E[T(x)]^{SRPT} \le 1/(1-\rho)$.

The proof of Theorem 4.4 follows immediately from Theorem 3.4, Theorem 4.2, and the following lemma, which allows us to bound the performance of SRPT by that under FB.

Lemma 4.1 *For all x and ρ, $E[T(x)]^{SRPT} \leq E[T(x)]^{FB}$.*

Proof : The proof is simply algebraic

$$
\begin{aligned}
E[T(x)]^{FB} &= \frac{x(1-\rho_x) + \frac{1}{2}\lambda E[{X_x}^2]}{(1-\rho_x)^2} \\
&= \frac{x}{1-\rho_x} + \frac{\frac{1}{2}\lambda\left(\int_0^x y^2 f(y)dy + x^2\overline{F}(x)\right)}{(1-\rho_x)^2} \\
&\geq \frac{x}{1-\rho(x)} + \frac{\frac{1}{2}\lambda\left(\int_0^x y^2 f(y)dy + x^2\overline{F}(x)\right)}{(1-\rho(x))^2} \\
&= \frac{x}{1-\rho(x)} + \frac{\frac{1}{2}\lambda\int_0^x y^2 f(y)dy + \frac{1}{2}\lambda x^2\overline{F}(x)}{(1-\rho(x))^2} \\
&\geq E[T(x)]^{SRPT}
\end{aligned}
$$

■

4.2.2 Intuition for dependence on load

Similarly to FB, notice that SRPT exhibits non-monotonicity under high load. Intuitively, this can be explained in the same way as it was for FB and PSJF in Section 3.3.2. Under high load, the large jobs in an SRPT system do not have the opportunity to increase their priority by reducing their remaining size. Thus, the largest job to arrive in a busy period will likely be the last to leave. This leads to unfairness.

However, SRPT does not always treat large jobs unfairly because during low load, the large job is often alone in its busy period, which provides it the opportunity to increase its priority as it receives service. Consequently, the large job will sometimes not be the last job to finish in the busy period.

4.3 Remaining size based policies

SRPT is one example of a remaining size based policy. In this section we will examine the entire class of remaining size based policies (i.e. policies where a job's priority is some function of its remaining size). The class of remaining size based policies includes many hybrid policies where, in order to minimize mean response time and curb the unfairness seen by large jobs, both jobs with very small and sufficiently large response times are given preferential treatment.

Note that, as in SRPT, we will choose to break ties among jobs in the system with the same priority according to FCFS. Thus, when multiple jobs have the same remaining size, and priority is inversely related to remaining size, then the job that has been in the system the longest will be worked on alone; however when multiple jobs have the same remaining size, and priority is directly correlated with remaining size, then the server will PS among the jobs.

Although SRPT is in this class and is Sometimes Unfair, not all such policies are Sometimes Unfair. For instance, the LRPT policy is Always Unfair as shown in Lemma 3.2.

Theorem 4.5 *All remaining size based policies are either Sometimes Unfair or Always Unfair.*

The remainder of this section will prove this theorem using the same method that was used in Section 3.4 and Section 3.2. We break the analysis into two cases: (1) the case when there exists a finite sized job that has the lowest priority and (2) when there is no finite sized job with the lowest priority.

Lemma 4.2 *Any remaining size based policy P with a finite remaining size C having the lowest priority is either Always Unfair or Sometimes Unfair.*

Proof : We will begin by deriving the expected performance seen by a job of original size C, entering the system under P. Notice that all work initially in the system will be completed before C begins to be worked on. In addition, all arrivals during this time that have size less than C will be completed before C leaves the system. However, once C starts being worked on and has remaining size t, the only arrivals that are guaranteed to finish before C leaves the system are those arrivals of size less than t. Thus,

$$E[T(C)]^P \geq \frac{\lambda E[X^2]}{2(1-\rho)(1-\rho(C))} + \int_0^C \frac{dt}{1-\rho(t)}$$

We will now show that C will be treated unfairly under high enough load. Using a similar derivation to that shown in Equations 4 and 5, we can see that $E[T(C)]^P > 1/(1-\rho)$ when

$$\frac{\lambda E[X^2]}{2(1-\rho)(1-\rho(C))} > \frac{C(\rho-\rho(C)) + \lambda m_2(x)}{1-\rho}$$

or, equivalently,

$$\frac{\lambda E[X^2]}{2(1-\rho(C))} - \lambda m_2(C) > C(\rho-\rho(C))$$

or, equivalently,

$$(1-\rho) + \left(\frac{\lambda E[X^2]}{2C(1-\rho(C))} - \frac{\lambda m_2(C)}{C}\right) > (1-\rho(C))\,.$$

Since $(1-\rho) \geq (1-\rho(C))$, we immediately see that P cannot be fair if $\rho(C) > \frac{1}{2}$. However, when C is the upper bound of a bounded distribution and $\rho = \frac{1}{2}$, the bound does not hold. In this case, we need to look at the system under a higher load. We can raise λ so that $\rho = \rho(C) > \frac{1}{2}$, in which case the bound holds.

When $\rho(C) < \frac{1}{2}$ we need to do a more detailed analysis. Since $\rho(C) < \frac{1}{2}$ we can raise λ so that $\rho = 2\rho(C)$. Notice that if this is not possible, it means that by raising λ we made $\rho(C) \geq \frac{1}{2}$, which we have already dealt with.

When $\rho = 2\rho(C)$, $E[X] = 2m_1(C) \stackrel{\text{def}}{=} 2\int_0^C tf(t)dt$. Further, this tells us that $E[X] - m_1(C) = m_1(C)$, but also $E[X] - m_1(C) = \int_C^\infty tf(t)dt$. Thus, $\int_0^C tf(t)dt = \int_C^\infty tf(t)dt$. Using this fact, we can notice that

$$\begin{aligned} E[X^2] = \int_0^\infty t^2 f(t)dt &= \int_0^C t^2 f(t)dt + \int_C^\infty t^2 f(t)dt \\ &\geq m_2(C) + Cm_1(C) \geq 2m_2(C) \end{aligned}$$

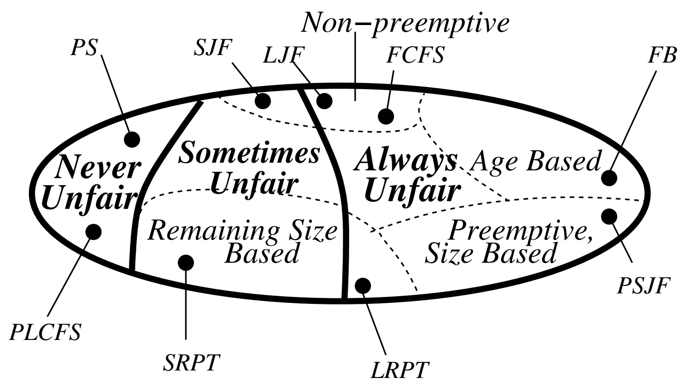

Figure 6: *Classification of unfairness proven in this paper.*

Thus, we can see that

$$(1-\rho) + \left(\frac{\lambda E[X^2]}{2C(1-\rho(C))} - \frac{\lambda m_2(C)}{C}\right) \geq (1-\rho) + \left(\frac{\lambda m_2(C)}{C(1-\rho(C))} - \frac{\lambda m_2(C)}{C}\right) > (1-\rho(C))$$

holds for all finite C. ■

Lemma 4.3 *Any remaining size based policy P where an infinitely sized job has the lowest priority is either Sometimes Unfair or Always Unfair.*

The proof of this final lemma follows from Theorem 4.3 and an argument symmetric to the proof of Lemma 3.5.

5 Conclusion

The goal of this paper is to classify scheduling policies in an M/GI/1 in terms of their unfairness. Very little analytical prior work exists on understanding the unfairness of scheduling policies, and what does exist is isolated to a couple particular policies. This paper is the first to approach the question of unfairness across all scheduling policies. Our aim in providing this taxonomy is, first, to allow researchers to judge the unfairness of existing policies and, second, to provide heuristics for the design of new scheduling policies.

In our attempt to understand unfairness, we find many surprises. Perhaps the biggest surprise is that for quite a few common policies, unfairness is a function of load. That is, at moderate or low loads, these policies are fair to all jobs. Yet at higher loads, these policies become unfair. This leads us to create *three* classifications of scheduling policies: Always Unfair, Sometimes Unfair, and Never Unfair (shown in Figure 1, repeated here for reference). Rather than classifying individual policies, we group policies into different types: size based, age based, remaining size based, and others. We prove that all size based and age based policies are Always Unfair, but that remaining size based policies and non-preemptive policies are divided between two classifications. The result that all size based policies are Always Unfair may seem surprising in light of the fact that one could choose to assign high priority to both small jobs and sufficiently large jobs in an attempt to curb unfairness.

With respect to designing scheduling policies, we find that under high load, almost all scheduling policies are unfair. However under low load one has the opportunity to make a policy fair by sometimes increasing the priority of large jobs. For example, PSJF and SRPT have very similar behavior and delay characteristics, but result in completely different unfairness classifications because SRPT allows large jobs to increase their priority, whereas PSJF does not.

A variety of techniques are used in order to classify policies with respect to fairness. For classifying individual policies it is useful to try to prove monotonicity properties for the policy over an interval of job sizes. It then suffices to consider the performance of the policy on just one endpoint of the interval. In classifying a group of policies, it helps to decompose the group into two cases: the case where the lowest priority job has a finite size/age, and the case where the lowest priority job has infinite size/age. In the latter case, we find that the fairness properties for the entire group of policies reduces to looking at one individual policy.

Since so many policies are Always Unfair, and so many others are Sometimes Unfair, it is interesting to ask *who* is being treated unfairly. Initially it seems that unfairness is an increasing function of job size, with the largest job being treated the most unfairly. This is in fact the case for bounded job size distributions. However, for unbounded job size distributions, we find this usually not to be the case. Instead, unfairness is monotonically increasing with job size up to a particular job size; and later is monotonically decreasing with job size. Thus the job being treated most unfairly ("top of the hump") is far from the largest. Interestingly, this "hump" changes as a function of load.

The above findings show that we are just beginning to understand unfairness in scheduling policies. This is a fertile area with many more properties yet to be uncovered.

References

[1] Baily, Foster, Hoang, Jette, Klingner, Kramer, Macaluso, Messina, Nielsen, Reed, Rudolph, Smith, Tomkins, Towns, and Vildibill. Valuation of ultra-scale computing systems. White Paper, 1999.

[2] N. Bansal and M. Harchol-Balter. Analysis of SRPT scheduling: Investigating unfairness. In *Proceedings of* Sigmetrics '01, 2001.

[3] M. Bender, S. Chakrabarti, and S. Muthukrishnan. Flow and stretch metrics for scheduling continous job streams. In *Proceedings of the 9th Annual ACM-SIAM Symposium on Discrete Algorithms*, 1998.

[4] Allen B. Downey. A parallel workload model and its implications for processor allocation. In *Proceedings of High Performance Distributed Computing*, pages 112–123, August 1997.

[5] M. Harchol-Balter and A. Downey. Exploiting process lifetime distributions for dynamic load balancing. *ACM Transactions on Computer Systems*, 15(3), 1997.

[6] M. Harchol-Balter, B. Schroeder, N. Bansal, and M. Agrawal. Implementation of SRPT scheduling in web servers. Technical Report CMU-CS-00-170, Carnegie Mellon University, 2000.

[7] M. Harchol-Balter, K. Sigman, and A. Wierman. Asymptotic convergence of scheduling policies with respect to slowdown. *Performance Evaluation*, 49(1-4):241–256, 2002.

[8] L. Kleinrock. *Queueing Systems*, volume II. Computer Applications. John Wiley & Sons, 1976.

[9] M. E. Crovella M. Harchol-Balter and S. Park. The case for SRPT scheduling in Web servers. Technical Report MIT-LCS-TR-767, MIT Lab for Computer Science, October 1998.

[10] R. Perera. The variance of delay time in queueing system M/G/1 with optimal strategy SRPT. *Archiv fur Elektronik und Uebertragungstechnik*, 47:110–114, 1993.

[11] I. Rai, G. Urvoy-Keller, and E. Biersack. FB: An efficient scheduling policy for edge routers to speedup the internet access. Unpublished manuscript.

[12] J. Roberts and L. Massoulie. Bandwidth sharing and admission control for elastic traffic. In *ITC Specialist Seminar*, 1998.

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

[14] Linus E. Schrage. A proof of the optimality of the shortest remaining processing time discipline. *Operations Research*, 16:678–690, 1968.

[15] F. Schreiber. Properties and applications of the optimal queueing strategy SRPT - a survey. *Archiv fur Elektronik und Uebertragungstechnik*, 47:372–378, 1993.

[16] A. Silberschatz and P. Galvin. *Operating System Concepts, 5th Edition*. John Wiley & Sons, 1998.

[17] W. Stallings. *Operating Systems, 2nd Edition*. Prentice Hall, 1995.

[18] A.S. Tanenbaum. *Modern Operating Systems*. Prentice Hall, 1992.

[19] A. Wierman, N. Bansal, and M. Harchol-Balter. A note comparing response times in the M/GI/1/FB and M/GI/1/PS queues. Technical Report CMU-CS-02-177, Carnegie Mellon University, September 2002.

[20] Ronald W. Wolff. *Stochastic Modeling and the Theory of Queues*. Prentice Hall, 1989.

A SRPT is Sometimes Unfair

We now complete the proof of Theorem 4.3.

Proof : To complete the proof, we need to bound $\rho(c_\infty)$. By showing that $\rho(c_\infty) < 1$ we illustrate a ρ_{crit} such that when $\rho \geq \rho_{crit}$ `SRPT` will lack slowdown monotonicity.

To understand what $\rho(c_x)$ is, we let $x > 0$ and notice

$$\begin{aligned}
\int_0^x \rho(x) - \rho(t)dt &\leq \int_0^x \frac{\rho(x)-\rho(t)}{1-\rho(t)}dt \leq \frac{1}{1-\rho(x)}\int_0^x \rho(x)-\rho(t)dt \\
1 &\leq \frac{\int_0^x \frac{\rho(x)-\rho(t)}{1-\rho(t)}dt}{\int_0^x \rho(x)-\rho(t)dt} \leq \frac{1}{1-\rho(x)}
\end{aligned}$$

So, c_x is such that

$$\begin{aligned}
\frac{1}{1-\rho(c_x)} &= \frac{\int_0^x \frac{\rho(x)-\rho(t)}{1-\rho(t)}dt}{\int_0^x \rho(x)-\rho(t)dt} \\
\rho(c_x) &= 1 - \frac{\int_0^x \rho(x)-\rho(t)dt}{\int_0^x \frac{\rho(x)-\rho(t)}{1-\rho(t)}dt}
\end{aligned}$$

Thus there exists a $\rho_{crit} < 1$ when the second term is bounded away from 1. The remainder of the proof bounds this value away from 1.

We continue by separating the integral in the denominator into three parts using r and s such that $\rho(r) = f\rho(x)$ and $\rho(s) = g\rho(x)$ for $f < g \in (0,1)$. Note that this is possible for some x under any non-constant service distribution.

$$\begin{aligned}
\int_0^x \frac{\rho(x)-\rho(t)}{1-\rho(t)}dt &= \int_0^r \frac{\rho(x)-\rho(t)}{1-\rho(t)}dt + \int_r^s \frac{\rho(x)-\rho(t)}{1-\rho(t)}dt + \int_s^x \frac{\rho(x)-\rho(t)}{1-\rho(t)}dt \\
&\le \frac{1}{1-\rho(r)}\int_0^r \rho(x)-\rho(t)dt + \frac{1}{1-\rho(s)}\int_r^s \rho(x)-\rho(t)dt \\
&\quad + \frac{1}{1-\rho(x)}\int_s^x \rho(x)-\rho(t)dt \\
&\stackrel{\text{def}}{=} \frac{1}{1-\rho(r)}A + \frac{1}{1-\rho(s)}B + \frac{1}{1-\rho(x)}C
\end{aligned}$$

Working with each of the pieces, we can derive

$$\begin{aligned}
A = \int_0^r \rho(x)-\rho(t)dt &= r\rho(x) - r\rho(s) + \lambda m_2(r) \\
&= r(1-f)\rho(x) + \lambda m_2(r) \\
&\to r(1-f)\rho + \lambda m_2(r) \text{ as } x \to \infty \\
B = \int_r^s \rho(x)-\rho(t)dt &= (s-r)\rho(x) - [s\rho(s) - \lambda m_2(s) - r\rho(r) + \lambda m_2(r)] \\
&= s(1-g)\rho(x) - r(1-f)\rho(x) + \lambda m_2(s) - \lambda m_2(r) \\
&\to s(1-g)\rho - r(1-f)\rho + \lambda m_2(s) - \lambda m_2(r) \text{ as } x \to \infty \\
C = \int_s^x \rho(x)-\rho(t)dt &= (x-s)\rho(x) - x\rho(x) + \lambda m_2(x) + s\rho(s) - \lambda m_2(s) \\
&= -s(1-g)\rho(x) + \lambda m_2(x) - \lambda m_2(s) \\
&\to -s(1-g)\rho + \lambda E[X^2] - \lambda m_2(s) \text{ as } x \to \infty
\end{aligned}$$

Further, we can notice that

$$\begin{aligned}
\lambda m_2(s) &= \lambda \int_0^r t^2 f(t)dt + \lambda \int_r^s t^2 f(t)dt \\
&\ge \lambda m_2(s) + r(\rho(s) - \rho(r)) \\
&= \lambda m_2(s) + r(g-f)\rho(x) \\
&= \lambda m_2(s) + r(1-f)\rho(x) - r(1-g)\rho(x) \\
&\to \lambda m_2(s) + r(1-f)\rho - r(1-g)\rho \text{ as } x \to \infty
\end{aligned}$$

Using this calculation in the formula for B, we see that as $x \to \infty$

$$\begin{aligned}
B &\ge (s-r)(1-g)\rho(x) \\
&\to (s-r)(1-g)\rho \stackrel{\text{def}}{=} \varepsilon
\end{aligned}$$

and

$$\begin{aligned} B &\leq s(1-g)\rho(x) + \lambda m_2(s) \\ &\to s(1-g)\rho + \lambda m_2(s) \stackrel{\text{def}}{=} \gamma \end{aligned}$$

Thus, for $N_x(A) \geq \frac{A}{\varepsilon}$ and $N_x(C) \geq \frac{C}{\varepsilon}$

$$\begin{aligned} N_x(A)B &\geq A \\ N_x(C)B &\geq C \end{aligned}$$

Calculating $N_\infty(A)$ we see

$$\begin{aligned} N_\infty(A) &\geq \frac{r(1-f)\rho + \lambda m_2(r)}{(s-r)(1-g)\rho} \\ &= \frac{r(1-f)}{(s-r)(1-g)} + \frac{\lambda m_2(r)}{(s-r)(1-g)\rho} \end{aligned}$$

and similarly for $N_\infty(C)$ we obtain

$$N_\infty(C) \geq \frac{-s(1-g)\rho + \lambda E[X^2] - \lambda m_2(s)}{(s-r)(1-g)\rho}$$

So, it is sufficient to have

$$\begin{aligned} N_\infty(A) &\geq \frac{r(1-f)}{(s-r)(1-g)} + \frac{\lambda E[X^2]}{(s-r)(1-g)\rho} \\ N_\infty(C) &\geq \frac{\lambda E[X^2]}{(s-r)(1-g)\rho} - \frac{s}{s-r} \end{aligned}$$

We now have bounds on the pieces of the integral. So, putting everything together we see that

$$\begin{aligned} \rho(c_\infty) &= 1 - \frac{\int_0^\infty \rho - \rho(t)dt}{\int_0^\infty \frac{\rho-\rho(t)}{1-\rho(t)}dt} \\ &\leq 1 - \frac{A+B+C}{\frac{1}{1-\rho(r)}A + \frac{1}{1-\rho(s)}B + \frac{1}{1-\rho}C} \\ &\leq 1 - \frac{A+B+C}{\frac{1}{1-\rho(r)}N_\infty(A)B + \frac{1}{1-\rho(s)}B + \frac{1}{1-\rho}N_\infty(C)B} \\ &\leq 1 - \frac{B}{\frac{1}{1-\rho(r)}N_\infty(A)B + \frac{1}{1-\rho(s)}B + \frac{1}{1-\rho}N_\infty(C)B} \\ &= 1 - \frac{1}{\frac{1}{1-f\rho}N_\infty(A) + \frac{1}{1-g\rho} + \frac{1}{1-\rho}N_\infty(C)} \\ &\stackrel{\text{def}}{=} 1 - \frac{1}{l} \end{aligned}$$

which tells us that as $x \to \infty$, $\frac{d}{dx}E[S(x)]^{SRPT} \to 0$ from below when

$$\rho \geq 1 - \frac{1}{2l}$$

The quantity l is bounded away from infinity as long as $s \neq r$, so there exists some $\rho_{crit} = 1 - \frac{1}{2l}$ where, for any service distribution, if $\rho \geq \rho_{crit}$ SRPT does not have slowdown monotonicity. ■

To better understand this proof it is interesting to look at the special case where $X \sim Exp(1)$. In this case, $f = \frac{1}{3}$, $g = \frac{2}{3}$, $E[X^2] = 2$, $s \approx \frac{2}{3}$, and $r \approx \frac{1}{3}$ (s and r are very approximate). So, we can calculate

$$\begin{aligned} N_\infty(A) &\geq \frac{2r}{(s-r)} + \frac{6\lambda E[X^2]}{(s-r)} \\ &\approx 38 \\ N_\infty(C) &\geq \frac{6\lambda E[X^2]}{(s-r)} - \frac{s}{s-r} \\ &\approx 35 \end{aligned}$$

and

$$\begin{aligned} l &\geq \frac{6}{5}N_\infty(A) + \frac{3}{2} + 2N_\infty(C) \\ &\approx 117.1 \end{aligned}$$

Which gives us that for $\rho > .99573$, SRPT will not have slowdown monotonicity under an $Exp(1)$ service distribution. Further, for these ρ, SRPT is guaranteed to treat some job size unfairly. It is important to point out the looseness of this bound. By plotting the actual equation for expected time in system under an $Exp(1)$ distribution we find that the true critical value for ρ in this case is just under .7, much lower than the value obtained using the method in the previous proof.