On Slowdown Variance as a Measure of Fairness

Maccio, Vincent J.
macciov@mcmaster.ca

Hogg, Jenell
hoggjm@mcmaster.ca

Down, Douglas G.
downd@mcmaster.ca

May 22, 2018

Abstract

When considering fairness one must ask two fundamental questions. Firstly, what does it mean to be fair? And secondly, how does one measure that fairness? Different authors have offered different notions and metrics to address these questions. We provide arguments identifying where past metrics fall short, discuss how the underlying motivations differ, and offer our own metric to address these issues. That is, we propose using a system's *slowdown variance* (SDV) as a measure for its fairness. Advantages of SDV are demonstrated via a suite of simulation experiments which compare a range of established policies under a range of service time distributions. These advantages include a decoupling of fairness from performance, an intuitive distinction between *last come first serve* and *processor sharing*, as well as recognition of starvation within *shortest remaining processing time*.

1 Introduction

Fairness in queueing systems has been an active topic in the stochastic modelling research community [19]. Perhaps the most fundamental questions in this domain are "What does it mean to be fair?" and "How can fairness be measured?". In the literature this is still a point of debate. That is, there does not exist a universally agreed upon definition of fairness. Moreover, different definitions of fairness may be in direct conflict or be of questionable applicability in certain contexts. We do not aim to determine which previously defined metrics are better than others. Rather, we offer a notion of fairness which is motivated by past definitions while looking to address some of the concerns these past metrics raise. Specifically, we propose using the slowdown variance (SDV) as a metric for fairness.

In order to understand the motivations behind our proposal, one must first understand how fairness has been presented in the research community. There are two fundamental views of fairness to consider: *temporal fairness* (also referred to as seniority-based fairness) and *proportional fairness*. Temporal fairness suggests that jobs should be served in order of their seniority. That is, if job J_1 arrives before job J_2, then job J_1 should be completed before J_2. A variety of fairness metrics such as politeness and order fairness provide means to measure a policy's temporal fairness. Our work does not discuss temporal fairness or its measures in any level of detail. For further reading, we direct the interested reader to [3,18]. Instead, our work focuses on, and is inspired by, the notion of proportional fairness. Proportional fairness posits that a job's response time should be proportional to its size. That is, informally, if a small job arrives shortly after a large job, it may be fair for the small job to complete before the large job does. In other words, the longer a job is going to take to serve the more fair it is that it waits in queue. For a comprehensive discussion and comparison of metrics pertaining to both temporal and proportional fairness, we direct the reader to [5].

Popular metrics for evaluating fairness are often times measures of the system's slowdown [6, 7, 10–12, 17, 20, 21]. Here, slowdown is defined as R/S, where R is the response time of a job and S is its size or service time (this work uses the two interchangeably for ease of exposition). The justification of this metric stems from the proportionality principle; it is fair that a job has a response time proportional to its service time (larger jobs should wait longer). It is worth noting that while researchers look to the slowdown as a base for fairness, slowdown is also considered a measure of performance; the response time is present in the numerator.

With regards to fairness, one of the first measures based off slowdown was simply examining the maximum slowdown [7]. This gives insight into the worst-case scenario and in turn bounds it in many cases, but does not give a picture of how fair a system is in the average case. To achieve a better overall understanding, authors began looking at forms of the the slowdown's expectation, i.e. $\mathbb{E}[R/S]$. The *conditional expected slowdown* [6] was introduced in response to growing interest in size-based policies that were suspected of treating larger jobs unfairly. It is defined as $\mathbb{E}[R(x)/x]$, where $R(x)$ is the expected response time for a job of size x. This metric provides finer-grained information than the expected slowdown. For example, it allows one to determine if certain job sizes are treated poorly.

Wierman and Harchol-Balter [20] leveraged the conditional slowdown to sort policies into categories. Based on the intuition that processor sharing (PS) is an inherently fair policy, they judge whether a policy is "Always Fair", "Sometimes Fair", or "Always Unfair". If a policy achieves a conditional expected slowdown that is less than or equal to the conditional expected slowdown offered by PS for all job sizes under any distribution it is Always Fair. If the conditional expected slowdown is less than or equal to that of PS for some job sizes under some distributions (but not all) it is Sometimes Fair. However, we find that this metric and criteria for evaluating proportional fairness have drawbacks. For one, the analysis in [20] is confined to an $M/G/1$ setting and the authors comment that it is unknown whether similar criteria can be derived for systems with general arrival processes and/or with multiple servers. Also, the categorization of policies into Always Fair, Sometimes Fair, and Always Unfair does not allow one to compare policies that fall into the same category, resulting in a partial ordering among policies.

In [13] Raz et al. proposed the Resource Allocation Queueing Fairness Measure (RAQFM), which was studied further by Avi-Izthak et al. in [4]. RAQFM is based off a *discrimination* measure, which in essence determines how far a policy is from giving each job an equal portion of the resource at every point in time. That is, it measures how much a policy differs from PS. This gives means to determine a total ordering of policies. However, this measure of fairness is still tightly coupled to the assumption that PS is perfectly fair. This assumption seems intuitive on the surface, but as will be seen later on in this work, can at times be problematic. Around the same time, Avi-Izthak et al. [2] also introduced the Slowdown Queueing Fairness measure (SQF). SQF, alongside its distinction from the SDV, is discussed in more detail in Section 2. In [2], it was proposed that a fair policy should offer every job a constant slowdown. The authors argued that the expected slowdown may not be sufficiently sensitive for evaluating fairness since policies can have similar expected slowdowns but have different behaviours about the expectation. The idea is that one can measure fairness with reference to an "ideally fair" slowdown.

Another means to measure fairness, the *discrimination frequency*, was introduced by Sandmann in [15] and discussed within the wider fairness framework in [16]. Here, a discrimination is counted every time one of the following events occurs: 1) a job arrives after another but completes before it, and 2) a job with greater remaining service time upon another job's arrival departs before the arriving job. Intuitively, the less frequently discriminations occur, the more fair the system is. Note, as is often the case with fairness, these two types of discrimination are in contention with each other. While the discrimination frequency is most certainly an attractive means to capture the tension between competing fairness properties, it belongs to a different school of thought than slowdown-based metrics (which is our focus).

As one can surmise there are several philosophies when measuring fairness. Moreover, fairness can be extremely sensitive to context. For example, what one deems to be fair for clients on a web-server, may be drastically different compared to what one deems to be fair for customers in a grocery store checkout. One notable difference between these two contexts is that in the grocery store checkout customers are free to see when other customers arrive, when they leave, how many items they are purchasing, etc. (white-box service), while on a web-server clients are often oblivious to how they are served relative to others (black-box service). This white-box/black-box differentiation can be seen in the fairness metrics as well. If for a given metric two policies with identical arrival and departure times for all jobs can be evaluated as having different levels of fairness then that metric is a white-box metric, otherwise, it is a black-box metric. Metrics like discrimination frequency and RAQFM are concerned with the details of how jobs are being served – how often jobs are being over taken, at a given point in time do all jobs have an equal share of the processor, etc; they are white-box metrics. On the other hand, metrics such as SQF, expected slowdown, and conditional slowdown are only concerned with the end performance, and have no regard as to how that performance is achieved; they are black-box metrics.

Our work's motivations are consistent with [2], but we offer a complementary viewpoint. We suggest that a truly fair policy would ensure that the ratio between a job's size and its response time remains constant (or as close to constant as possible). Therefore, the closer the SDV is to zero, the more fair a policy is.

As observed in [2], we shall see that using this basis for fairness will provide insights that using expected or conditional slowdown would otherwise not capture. Most notably, using the expected slowdown or the conditional slowdown is known to equate two well-known policies, processor sharing and last come first serve, in terms of their fairness. However, we find evidence that SDV will determine one to be more fair than the other, as will be discussed in Section 3. Furthermore, since SDV is a black-box metric, for consistency of context and application this work focuses on comparing it against other black-box metrics.

Any mention of SDV in the literature has been brief – to our knowledge no such study or presentation of SDV exists. One possible explanation for this absence is that analytic expressions for SDV are difficult to determine (this appears to be a fundamental problem, as we have made attempts to generate analytic results). As such, we sacrifice analytic tractability in order to examine these ideals of fairness empirically. The contributions of this paper include, but are not limited to,

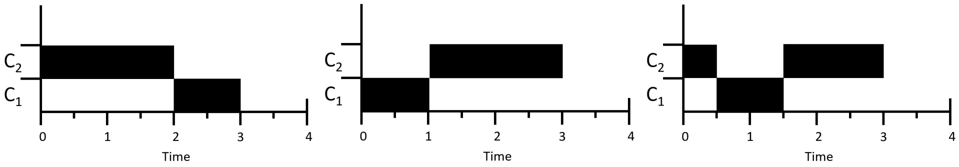
1. The introduction and justification of using SDV as a fairness metric, found in Section 2.
2. An extensive simulation study pertaining to SDV and expected slowdown under different scheduling policies and distributions, found in Section 3.
3. Several key observations and insights into the behaviour of SDV across these different configurations, also found in Section 3, which enriches the overall discussion of fairness.

2 Definitions and Justification

As stated previously when discussing fairness, there are two important aspects which must be made clear. Firstly, what does it mean to be fair? And secondly, how does one quantify fairness? As seen in Section 1 these are subjective issues. Nevertheless, we believe metrics for fairness exist which are grounded in intuition, and moreover, such metrics are independent/decoupled from performance.

We proceed by addressing the broad but fundamental question of what it means to be fair. Consider a scenario where each customer C has exact knowledge of its size, or service time, denoted by S_C, as well as its response time, denoted by R_C, but no knowledge of *how* it is served. Suppose that customer C_1 has a service time of one minute, $S_{C_1} = 1$, and a response time of three minutes, $R_{C_1} = 3$. With no information about other customers, they may simply perceive this as *typical* system performance. When a second customer C_2 is introduced, things become more complicated. Continuing with the example, let $S_{C_2} = 2$ and $R_{C_2} = 2$, so that $S_{C_1} < S_{C_2}$, but $R_{C_1} > R_{C_2}$. Therefore, we argue that even an impartial onlooker would view C_1 as being treated unfairly, as it requires less system capacity yet has a longer response time. As such, C_1 is likely to be dissatisfied. This dissatisfaction stems from C_1's expectation of treatment relative to others. That is, it stems from C_1's notion of fairness.

In this small example it is intuitive that all parties would agree the system is not fair. However, when *is* the system fair? A perfectly fair system can be demonstrated by considering the previous example where the response times vary according to the scheduling policy implemented. Assume the previous example was implementing the policy denoted by π_1. That is, under π_1, $R_{C_1} = 3$ and $R_{C_2} = 2$. Under another policy, denoted by π_2, suppose that $R_{C_1} = 1$ and $R_{C_2} = 3$. Lastly, under policy three, denoted by π_3, suppose that $R_{C_1} = 1.5$ and $R_{C_2} = 3$. Assuming both customers arrive at the same instant, a realization of these three policies is illustrated in Figure 1, and the values are summarized in Table 1.



(a) π_1 (b) π_2 (c) π_3

Figure 1: Time-lines of the three policies, black portions represent when a customer is receiving service.

Looking at these policies in greater detail allows one to begin formalizing fairness. As discussed previously, π_1 is an intuitively unfair policy. Inspecting π_2 one discovers some positive aspects. Specifically, the customer

Policy	R_{C_1}	R_{C_2}	R_{C_1}/S_{C_1}	R_{C_2}/S_{C_2}
π_1	3	2	3	1
π_2	1	3	1	1.5
π_3	1.5	3	1.5	1.5

Table 1: Summary of metrics associated with C_1 and C_2

with the larger size experiences a longer response time than the customer with the smaller size. This would seem to agree with what most would consider fair. A natural next question is if one has a longer service time, how much longer should one have to wait? The answer to this question is, unsurprisingly, dependent on the experience of the other customers, or customer in this case. Under π_2, C_1 has a response time equal to its service time, while C_2 has a response time greater than its service time. It is our stance that such a configuration is inherently unfair. In π_1, C_1 is unfairly treated, while in π_2, C_2 is unfairly treated. With this in mind we move our focus to π_3. Similar to π_2, under π_3 the customer with a longer service time, C_2, also has a longer response time. Here however, the response time of C_1 does not equal its service time. That is, $R_{C_1} > S_{C_1}$. The response time of C_1 is 1.5 times greater than its service time, i.e. $R_{C_1}/S_{C_1} = 1.5$. With respect to fairness, we now have a benchmark to compare against. That is, customers do not necessarily perceive the system as unfair if they experience longer than typical response times, but rather they perceive it as unfair if their response time is inordinately large relative to their service time (compared to the experience and size of other customers). Here, $R_{C_1}/S_{C_1} = R_{C_2}/S_{C_2}$. In other words, both customers have equal slowdowns. From here, fairness can formally be defined.

Definition 1. ***Fair:*** *A system is said to be perfectly fair if, and only if, for all jobs J_i and J_k, $R_{J_i}/S_{J_i} = R_{J_k}/S_{J_k}$. In other words, a system is perfectly fair if the slowdown is constant.*

There are a few things to note regarding this definition. Unlike some other definitions of fairness, ours is completely decoupled from performance (to the best of our knowledge, Avi-Itzhak et al. [2] is the only other work where such decoupling is present). That is, a system can be perfectly fair while having poor performance. This can be seen in the extreme case where the system never processes jobs. A less extreme case can be seen by viewing π_2 and π_3 in Figure 1. Here C_1 has a lower response time under π_2 than under π_3, while at the same time C_2 has equal response time under both policies. Therefore, if performance were the only concern π_2 would be the better policy. Under Definition 1, π_2 is not perfectly fair, while π_3 is. There is no denying that π_2 would be preferable to π_3 (from a performance viewpoint), as every job does at least as well under π_2. The crucial point for our work is that we are defining the fairness of a policy in isolation, i.e. jobs under policy π_3 only know how well other jobs do under π_3, not how well they would do under π_2. So, π_3 is seen as fair, but has sub-optimal performance. This example should serve to reinforce the notion of the decoupling of fairness from performance.

Due to the stochastic nature of these systems, i.e. random arrival times, job sizes, etc., a perfectly fair policy (which is also stable) in general does not exist. However, using this definition as an ideal to strive towards, it would seem one should be able to reason that one policy is more fair (or less unfair) than another. For small systems and toy examples, like the ones presented earlier in this section, it is clear which policies are preferred, if being fair is the ultimate goal. However, when these systems are viewed over longer periods of time with many events occurring, it becomes harder to rely solely on intuition to determine which policy is more fair. And where it may be easy to find specific examples of clearly fair or unfair sample paths, one must also consider the aggregate behaviour. Therefore, to compare the fairness of policies, it is imperative to have a metric. The metric we propose to measure the fairness of a system is the slowdown variance (SDV), i.e. $\text{Var}(R/S)$.

The first and perhaps most important property of SDV to note is that by Definition 1, if a system is perfectly fair then SDV is zero, i.e. $\text{Var}(R/S) = 0$. Moreover, from the definition of variance the higher the SDV is the further the system is from being perfectly fair. In other words, SDV provides a means to rank the fairness of a set of policies. Returning to our example, without using a metric such as SDV, the most we could say about the three policies is that π_3 is perfectly fair, while π_1 and π_2 are not. Intuitively, π_2 is more fair than π_1; it is more fair for the longer job to wait behind the shorter one. When the SDVs of these two policies are calculated and used as a measure of fairness, it agrees with our intuition that π_2 is more fair than π_1. That is, letting $\text{Var}(R/S)^{\pi}$ denote the SDV under policy π, $\text{Var}(R/S)^{\pi_2} < \text{Var}(R/S)^{\pi_1}$.

	Average Response Time	Average Slowdown	SDV
π_1	6.5	1.40	0.320
π_2	6.5	1.45	0.005

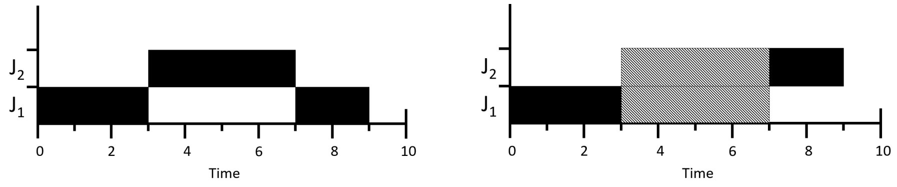
Table 2: Example of ESD vs SDV

2.1 Comparing Slowdown Variance to Other Fairness Metrics

Having given our definition of what it means to be fair, and furthermore, how to measure that fairness, it is instructive to compare these ideas against existing fairness metrics. As seen in Section 1 there are many different metrics to compare against. With that in mind we feel our metric is most closely related to SQF and other slowdown metrics. Perhaps the most natural metric to compare SDV against is the expected slowdown (ESD). The first major differentiation is that, as mentioned before, SDV is primarily concerned with fairness, while ESD is coupled to the performance of the system. Moreover, while ESD incorporates the notion that larger jobs should wait longer, it lacks a strict definition of perfect fairness.

Even with just these distinctions in mind, it is not surprising that these two metrics often disagree on which policy is more fair. A valid question remains: does ESD disagree with SDV only due to its coupling to performance? In other words, if $\mathbb{E}[R/S]^{\pi_1} \leq \mathbb{E}[R/S]^{\pi_2}$ and $\mathrm{Var}(R/S)^{\pi_2} < \mathrm{Var}(R/S)^{\pi_1}$ must it then be true that $\mathbb{E}[R]^{\pi_1} < \mathbb{E}[R]^{\pi_2}$? To show that this is not *always* the case, we offer a counterexample. Consider two policies π_1 and π_2, each of which serves two jobs, denoted by J_1 and J_2. Let the sizes and arrival times of J_1 and J_2 be denoted by $S_1 = 5$ and $S_2 = 4$, and $a_1 = 0$ and $a_2 = 3$, respectively. For the counterexample to be valid, it must hold that $\mathrm{Avg}(R/S)^{\pi_1} \leq \mathrm{Avg}(R/S)^{\pi_2}$ and $\mathrm{Var}(R/S)^{\pi_2} < \mathrm{Var}(R/S)^{\pi_1}$ (the metrics disagree on which policy is the fairer of the two), while $\mathrm{Avg}(R)^{\pi_1} \geq \mathrm{Avg}(R)^{\pi_2}$ (the policy which ESD favours, does not have a smaller expected response time). That is, the ESD under π_1 is not lower due to its coupling to performance, but rather ESD and SDV disagree strictly due to their conflicting notions of fairness.

Following through with the example, let the service discipline of π_1 and π_2 follow that illustrated in Figure 2. That is the departure times of J_1 and J_2 are 9 and 7, and 7 and 9, under policies π_1 and π_2 respectively. The corresponding values of interest can be seen in Table 2. Here it is seen that SDV and ESD disagree on which policy is more fair (ESD favours π_1 while SDV favours π_2), but the average performance of the two policies is equal. Therefore, the disagreement of SDV and ESD cannot be attributed to ESD coupling to performance, and must instead be attributed to their underlying notions of fairness.



(a) π_1 execution path

(b) π_2 execution path

Figure 2: Time-lines of π_1 and π_2, black portions represent when a job is receiving service, while striped/grey portion represent a job is receiving shared service.

Although this is only a small example, the results can be extended to a more general case. The astute reader may have noticed that π_1 and π_2 are following the well known policies LCFS and PS, respectively. For an $M/G/1$ queue it is known that $\mathbb{E}[R]^{\mathrm{LCFS}} = \mathbb{E}[R]^{\mathrm{PS}}$ and furthermore $\mathbb{E}[R/S]^{\mathrm{LCFS}} = \mathbb{E}[R/S]^{\mathrm{PS}}$ [10]. Note that while the values of the average slowdown differ in Table 2, this is only across one potential sample path and not a contradiction of the equality of the expectations. It will also be seen in Section 3 that our simulation results strongly suggest that $\mathrm{Var}(R/S)^{\mathrm{PS}} < \mathrm{Var}(R/S)^{\mathrm{LCFS}}$. In other words, if ESD is used as a metric for fairness, LCFS would be deemed just as fair as PS, which we argue would contradict typical intuition. However, as will be seen in Section 3, SDV avoids this issue.

As mentioned in Section 1, another potential metric for fairness is the Slowdown Queueing Fairness (SQF) measure from [2]. This metric was formulated using similar arguments we present to justify SDV. That is, all jobs should have equal slowdown. Of course the question remains, what is the difference between these two metrics, and why would one use one over the other?

To guide our discussion, we first present the definition of SQF:

$$SQF = \mathbb{E}[(R - (\mathbb{E}[R]/\mathbb{E}[S])S)^2] = \mathbb{E}[R^2] - (\mathbb{E}[R]/\mathbb{E}[S])\mathbb{E}[RS] + (\mathbb{E}[R]/\mathbb{E}[S])^2\mathbb{E}[S^2].$$

For comparison, the definition of SDV is:

$$SDV = \mathbb{E}[(R/S)^2] - \mathbb{E}[R/S]^2.$$

One noticeable difference between these two definitions is that SQF depends directly on the second moment of the service time distribution, whereas SDV depends on the second moment of the slowdown. One may see this as a small difference but the implications are drastic. Suppose that the service time distribution is such that $\mathbb{E}[S^2]$ is large (the Pareto distribution for example). In this case, $(E[R]/E[S])^2E[S^2]$ can become the dominating term for SQF, since $1 \leq (E[R]/E[S])^2$. This becomes more problematic once one notes that $E[S^2]$ is independent of the employed policy, resulting in fairness being insensitive to the choice of policy. On the other hand, all terms present in SDV, dominant or not, are dependent on the employed policy. Moreover, if the second moment of the service time distribution is infinite, policies become incomparable to each other, i.e. they are all equally unfair. This observation is even more troublesome once one realizes that distributions in which fairness is a concern are more often than not distributions where the service time variance is extremely high. Therefore, we argue that under the distributions of interest, i.e. heavy tailed, decreasing failure rate, etc., SQF could be a problematic choice. This issue could be one of the reasons [2] limited their experiments to non-heavy tailed distributions.

While the above criticism is certainly a concern, we would be remiss if we did not point out a notable advantage SQF has over SDV. SQF leads to closed form expressions in the case of an $M/G/1$ queue, whereas for SDV, analytic results are difficult to arrive at. In fact, to the best of our knowledge an expression for the slowdown variance of an $M/G/1$ queue under any scheduling policy remains unknown.

3 Simulation and Discussion

The SDV of a system is difficult to analyse directly, even for single server systems implementing well known policies. As such, to examine the effects of policy choice on SDV, and in turn system fairness, we rely on simulation. All simulations are of $M/G/1$ queues, where we varied the system's service time distribution, policy, and load, using basic Matlab libraries. Each simulation was run for 10,000,000 arrivals and departures. In most cases this was sufficient to observe convergent behaviour. The one exception to this were the simulations for Pareto distributed service times, which were run for 1,000,000 arrivals. We are aware that in general one would wish to run simulations with underlying Pareto distributions for a longer period of time to allow for convergent behaviour, however, due to the nature of these distributions (larger jobs remaining in the system longer), runtime complexity required us to limit the number of arrivals. To verify the simulations, all known analytic results we are aware of were sanity checked. For example, the expected response time of PS and LCFS are extremely close to equal for all simulations, SRPT always has the lowest expected response time, LAS has the lowest expected response time among all blind policies for Pareto service times [8], FCFS and P-ROS have extremely close to equal expected response times for exponential service time, etc.

To grant the reader a comprehensive understanding of the simulation results, for each service time distribution we present the expected response time, response time variance, expected slowdown, and slowdown variance. It is worth noting that the y-axes on the graphs corresponding to slowdown/response time variance are logarithmic. One may also note the exclusion of the aforementioned SQF metric from our experiments. We would conjecture that SDV fairness results would be comparable to those of SQF for low variance distributions, while for high variance distributions, SQF is problematic (as discussed in Section 2). Therefore, we narrow our discussion to a focused comparison of SDV to ESD. While not written under the intent of public use, all source code can be found at [1]. Moreover, as mentioned in Section 1, we chose to limit our discussion and simulations to black-box metrics.

3.1 Policies

Although most policies which we simulate are well known, for the sake of completeness we give their definitions in full.

FCFS (First Come First Serve): Jobs are served in the order that they arrive.

LCFS (Last Come First Serve - Preemptive Resume): Jobs are scheduled in order of how recently they arrived to the system. If a job arrives while another is being processed, the job being processed is placed back into the queue and the most recent arrival is processed.

PS (Processor Sharing): All jobs are given equal share of the processor's capacity.

LAS (Least Attained Service): The job with the least attained service gets priority. If there is more than one job with the least attained service, they are processed using processor sharing. Note, this policy is also referred to as as Foreground-Background Processing.

P-ROS (Random Order of Service - Preemptive Resume): Jobs are served in random order. If a job is being processed when a new job arrives, the job being processed is preempted and placed back into the queue. A job is then randomly selected from the queue.

SRPT (Shortest Remaining Processing Time): Jobs are scheduled in order of remaining service time. If a job arrives that has a smaller service time than the one currently being processed, the job in process is preempted by the arriving job and placed back into the queue. If a job is placed back into the queue, no accumulated service is lost (preemptive resume).

FSP (Fair Sojourn Protocol): Jobs are scheduled in the order in which they would complete in a processor sharing queue.

While most of the above policies are well known in the queueing literature, FSP deserves some special attention. It is known that under FSP every job will depart at least as soon as it would under PS [9]. In turn this makes an interesting case when comparing the fairness of these two policies (FSP and PS). FSP is strictly improving performance, while at the same time no individual is doing worse (relative to when they would depart under PS). However, individuals may do worse relative to others, potentially leading to an unfair system overall. This is a general case of the first example given in Section 2, where π_3 is the most fair policy, but π_2 has strictly better performance.

The simulation results are sorted into three broad categories, first those with increasing failure rates (IFR distributions), second those with decreasing failure rates (DFR distributions), and lastly specific distributions which offer some additional insights.

3.2 Results for IFR Distributions

Within the set of IFR distributions, we begin with the simplest: constant service times. Here, an observation immediately worth noting is that a number of policies become equivalent. Specifically, FCFS, SRPT, and FSP will process jobs in the same order. It is known that FCFS will minimize the response times when the service time distributions have an increasing failure rate amongst policies that do not use size information (blind policies) [14], i.e. FCFS, LCFS, P-ROS, LAS, and PS. This leads to the well known result that when service times are constant, FCFS will yield optimal slowdown amongst blind policies. What may not be as intuitive is how FCFS (and therefore SRPT and FSP) would perform when it comes to fairness. It would be reasonable to hypothesize that a policy such as PS would be more fair than FCFS in this case, since at a glance PS tries to be fair at every point in time. This is not the case however, as can be seen in Figure 3-(d), FCFS has lower SDV than PS for all simulated loads. This is somewhat surprising since PS is thought to be fair by construction (as discussed in Section 1). One explanation for this is the following. Consider jobs which arrive to an empty system. Under FCFS, they have equal response times and slowdowns. Therefore, conditioning on jobs which arrive to an empty system, the SDV is zero. Under PS another job could arrive, causing the job currently in service to wait longer than it would have otherwise. This argument can be extended to finding an arbitrary number of jobs upon arrival. That is, if a job arrives, and no others arrive until said job is complete, the response time of that job is equal under FCFS and PS. When one begins to condition on jobs arriving after it however, the response time for PS changes (inducing more variance), while under FCFS the response time is independent of subsequent arrivals. An implication of this behaviour is that doing something which is viewed as fair at a given time point (such as PS), can result in an unfair system. Overall this is a positive result, since it shows performance and fairness are not in direct contention with each other. In other words, the policy which has the best performance, may also be the most fair.

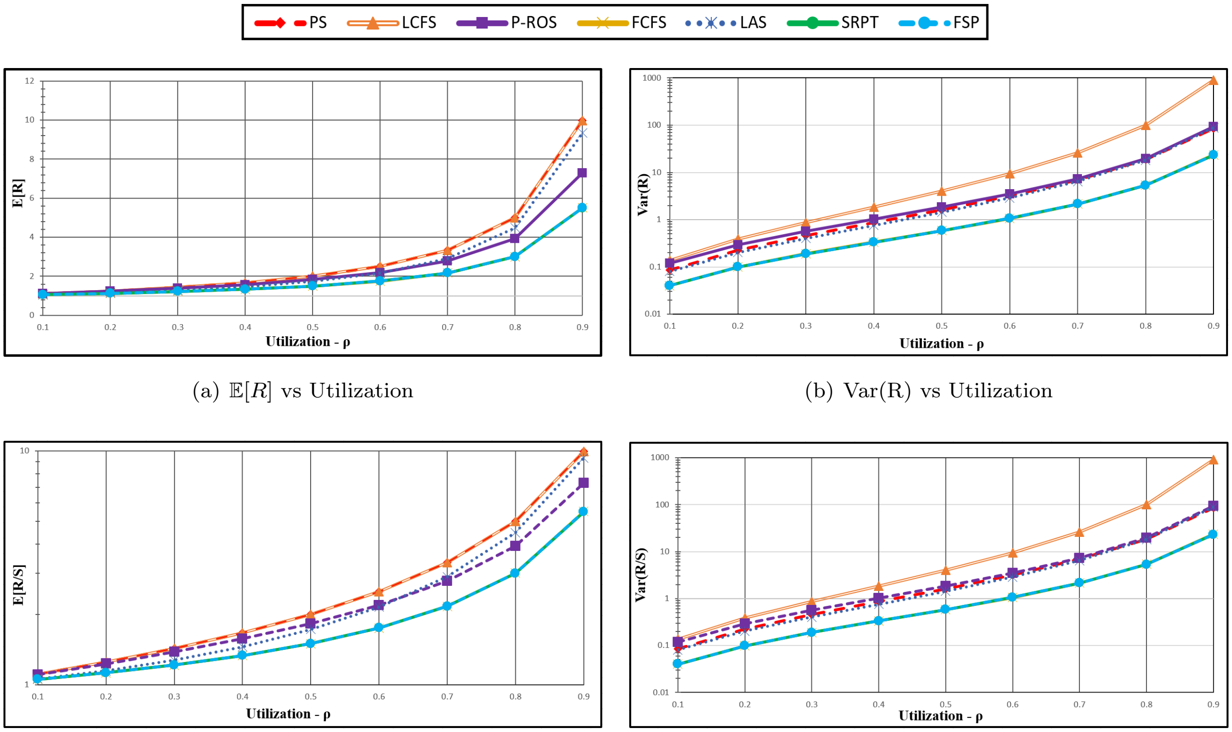
Observation 1. *It is possible for a policy to have the best performance, while being the most fair. As an example, across all simulated policies, when service times are constant, FCFS, SRPT, and FSP have the best performance while also being the most fair.*

For the remainder of the IFR simulations, SRPT and FSP continue to perform well, although not necessarily the best, while in some cases FCFS performs poorly. This is not unexpected however, as policies such as SRPT and FSP use service time information, while policies such as FCFS, LCFS, PS, and LAS are blind. This is demonstrated in Figures 4 and 5.

Under the Erlang-2 system we see that there is a difference in the rankings of policies in terms of ESD and SDV. It was shown in [8] that for Erlang-2 systems LAS will yield optimal slowdown despite maximizing the expected response time and one can observe this in our results (Figure 4). Despite performing well in expectation, LAS is outperformed at higher utilizations in terms of SDV by PS, i.e. its ranking is load dependent. This load dependent ordering is not a product of large jobs rarely receiving service, i.e. starvation, since FSP mimics the service order of PS which always gives large jobs at least part of the total processing power. Rather, this is due to small jobs experiencing unfair performance i.e. inordinately small response times. When the load becomes large, the expected number of jobs in the system becomes large in turn. As such, even under a policy which avoids starvation, such as PS, large jobs would expect to have large (albeit reasonable) response times. Moreover, under a policy such as PS, the penalty for having a large number of jobs in the system would also be felt by the small jobs. This is the distinguishing factor when comparing PS to FSP. While the large jobs are not being starved, it is true the large jobs take on the entire penalty of having many jobs in the system. This leads to small jobs getting a *free pass* through the system under FSP. This in turn leads to a high SDV and an unfair system. It remains curious though, that SRPT still does well here from a fairness standpoint, as it would seem a similar argument could be applied, implying it should also be unfair. One explanation for this is SRPT is not constrained to worry about starvation, which interestingly gives it the freedom to be more fair overall. Due to the distribution being IFR, extremely large jobs are not likely to appear. Thus, perhaps this disregard of starvation is justified in this setting. With all that in mind, we leave the specifics of this behaviour for this specific distribution as an open question. Other issues regarding SRPT and starvation will be seen in Section 3.3.

Observation 2. *For two policies π_1 and π_2, the ordering of fairness can be load dependent.*

Under uniformly distributed service times, a similar but reversed effect is observed. That is, FSP performs the best and is the most fair, while SRPT begins to become unfair as the load increases. We argue this is due to the lower bound ($a = 0.5$) imposed on the uniform distribution. Unlike with the Erlang-2 distribution, jobs cannot be extremely small compared to the mean. In fact, the bound on the distribution limits the largest possible job to be only $b/a = 1.5/0.5 = 3$ times larger than the smallest possible job. As such, when SRPT gives a job priority over another, it is known that the now highly prioritized job cannot be drastically smaller than the now low priority job. Worse yet, if a slightly smaller job arrives while the low priority job is waiting to receive service, that job is guaranteed to depart before the one which was waiting. On the other hand, because FSP employs the departure order of PS, it may still be the case that a job may depart before a smaller one, if it was waiting longer. Taking these ideas into account under a uniform distribution it becomes reasonable to expect SRPT will become relatively unfair at higher loads, while at higher loads FSP becomes the most fair. Given the fact that the uniform distribution has low variance, previous observations for FCFS under constant service times remain applicable.



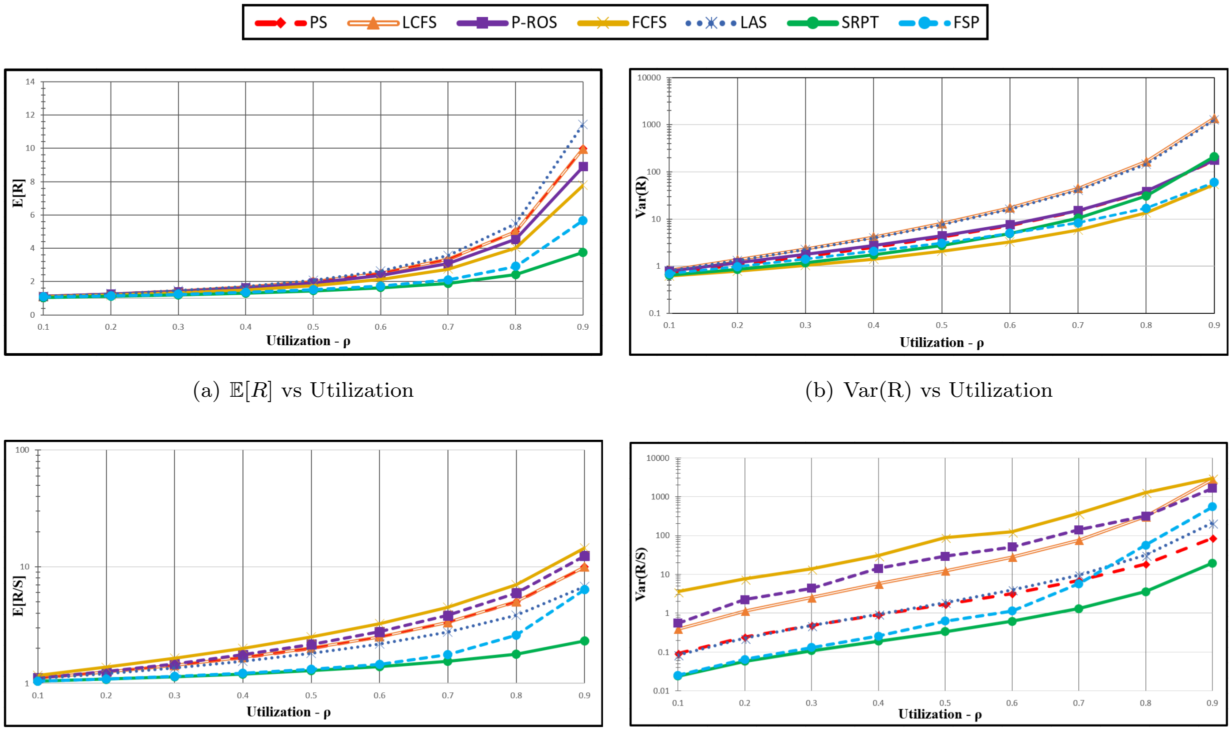
(a) $\mathbb{E}[R]$ vs Utilization

(b) Var(R) vs Utilization

(c) $\mathbb{E}[R/S]$ vs Utilization

(d) Var(R/S) vs Utilization

Figure 3: Simulation results for system where service times are constant: $a = 1$.



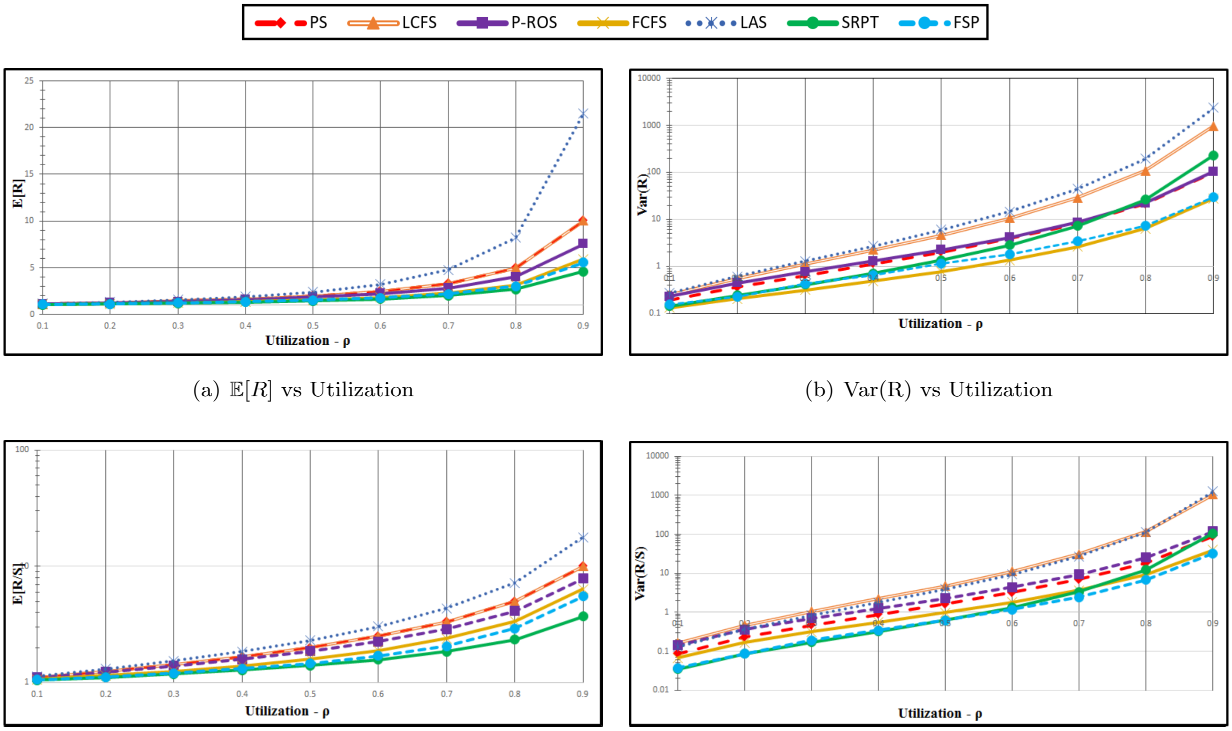
(a) $\mathbb{E}[R]$ vs Utilization

(b) Var(R) vs Utilization

(c) $\mathbb{E}[R/S]$ vs Utilization

(d) Var(R/S) vs Utilization

Figure 4: Simulation results for systems where service times follow an Erlang distribution: $k = 2$, $\mu = 0.5$



(a) $\mathbb{E}[R]$ vs Utilization

(b) Var(R) vs Utilization

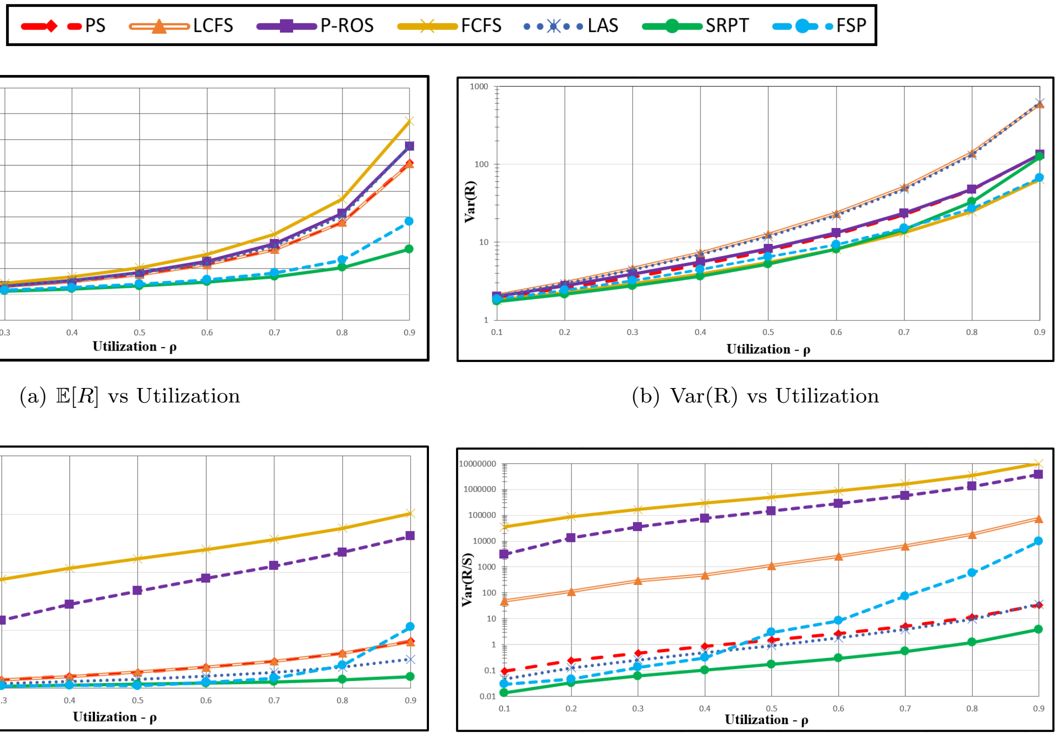
(c) $\mathbb{E}[R/S]$ vs Utilization

(d) Var(R/S) vs Utilization

Figure 5: Simulation results for systems where service times follow a uniform distribution: $a = 0.5$, $b = 1.5$.

3.3 Results for DFR Distributions

Arguably the more interesting and problematic distributions are those which are DFR. This is because such distributions are apt to produce massive jobs relative to their means. This provides two broad challenges. First, if jobs cannot be preempted, slowdown is considerably increased, as many small jobs can get stuck behind one large job. Second, if jobs can be preempted, many small jobs could overtake the large job in priority, leading to large job starvation. We simulated hyper-exponential and Pareto distributions. It is worth noting that in the figures of this section, some curves are absent due to known behaviours which lead to results that are orders of magnitude worse than the majority of the policies. For example, the expected response time of FCFS under the Pareto distribution is infinitely large. Furthermore, in Figure 6 job sizes are given a lower bound; the reasoning behind this is discussed in Section 3.4.



(a) $\mathbb{E}[R]$ vs Utilization

(b) Var(R) vs Utilization

(c) $\mathbb{E}[R/S]$ vs Utilization

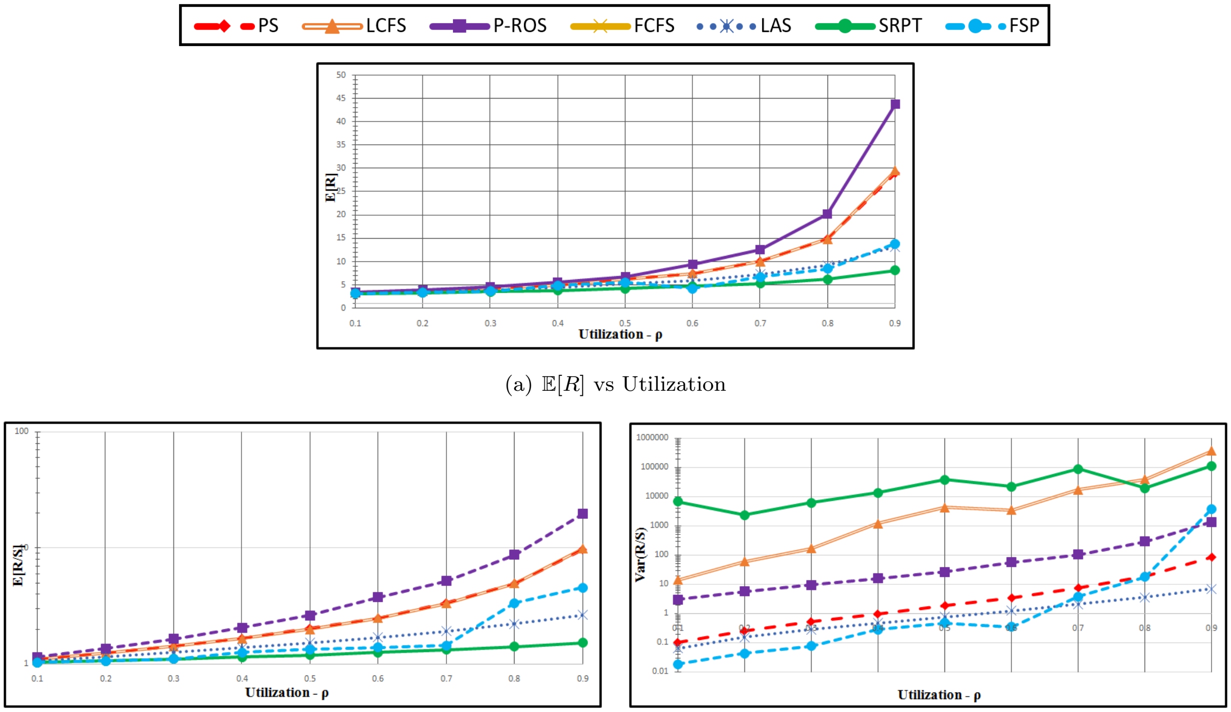
(d) Var(R/S) vs Utilization

Figure 6: Simulation results for systems where service times follow a bounded hyper-exponential distribution; $p_1 = 0.75$, $p_2 = 0.25$, $1/\mu_1 = 1.24$, $1/\mu_2 = 0.24$, and lower bound of 0.01, where p_i denotes the probability of sampling from an exponential distribution with corresponding rate μ_i

As with the IFR distributions, SRPT and FSP are positive policies from the viewpoint of performance. It is well known that SRPT minimizes all moments of the response time under all distributions for single server systems. Furthermore, if one were using a traditional fairness metric, such as expected slowdown, it would be difficult to make a case to employ any policy other than SRPT, as SRPT also has the lowest expected slowdown. Another well known property of SRPT, however, is that it starves large jobs. As argued in the previous section, starvation is unfair. Therefore, one would expect to see SRPT perform poorly under a fairness metric, especially under a heavy-tailed distribution where these starvation issues are amplified. SDV is consistent with this notion. In Figure 7-(d) not only does SRPT fail to be the most fair, but is one of the least fair policies simulated. This of course arises due to the amount of time large jobs must wait before they depart. In contrast, under SRPT, an extremely small job is likely to have a slowdown of one, the minimum achievable. That is, SRPT is unfair in two respects; it gives both an unfair disadvantage to large jobs and an unfair advantage to small jobs. Hence SDV captures inherent unfairness which SRPT invokes, whereas the expected slowdown not only fails to capture the issue of starvation, but goes further by implying SRPT to be the most fair of all simulated policies.

When it comes to FSP, the expected slowdown does capture some of the unfair qualities as the load increases, as was previously discussed for IFR distributions. These unfair tendencies are amplified in the SDV, where under heavy loads PS, P-ROS, and LAS are all deemed more fair than FSP. Looking to the ESD, only LAS (and SRPT) are deemed more fair than FSP. It still remains then, that while FSP and SRPT may be positive choices from a performance standpoint, they may be extremely unfair.

Observation 3. *While the non-blind policies SRPT and FSP seem like attractive choices from the viewpoints of $\mathbb{E}[R]$ and $\mathbb{E}[R/S]$ they could be extremely unfair from the viewpoint of Var(R/S). That is, Var(R/S) captures the unfairness of large jobs being starved, and small jobs receiving an unfair advantage.*



(a) $\mathbb{E}[R]$ vs Utilization

(b) $\mathbb{E}[R/S]$ vs Utilization

(c) Var(R/S) vs Utilization

Figure 7: Simulation results for systems where service times follow a Pareto distribution: $\alpha = 1.5$, $x_m = 0.5$, where α and x_m are the shape and scale parameters, respectively. Note that the graph of Var(R) vs. ρ has been excluded due to the high variance of the Pareto distribution.

The previous observation does not imply that if one wishes to be fair under the Pareto distribution then one must sacrifice performance. Here, one sees favourable results for LAS. Firstly, for DFR distributions, LAS is known to be an optimal blind policy. While SRPT and FSP slightly outperform it, LAS remains competitive with access to significantly less information. Moreover, when one examines the SDV of LAS it is seen that it also is the fairest of all blind policies, and for high loads, it is the fairest of all policies, blind or otherwise.

Observation 4. *For all DFR experiments, of all the blind policies, LAS achieves the lowest expected response time, while remaining the most fair, i.e. achieving the lowest Var(R/S). Furthermore at high loads, LAS is the most fair over all simulated policies.*

3.4 M/M/1 and Bounded Distributions

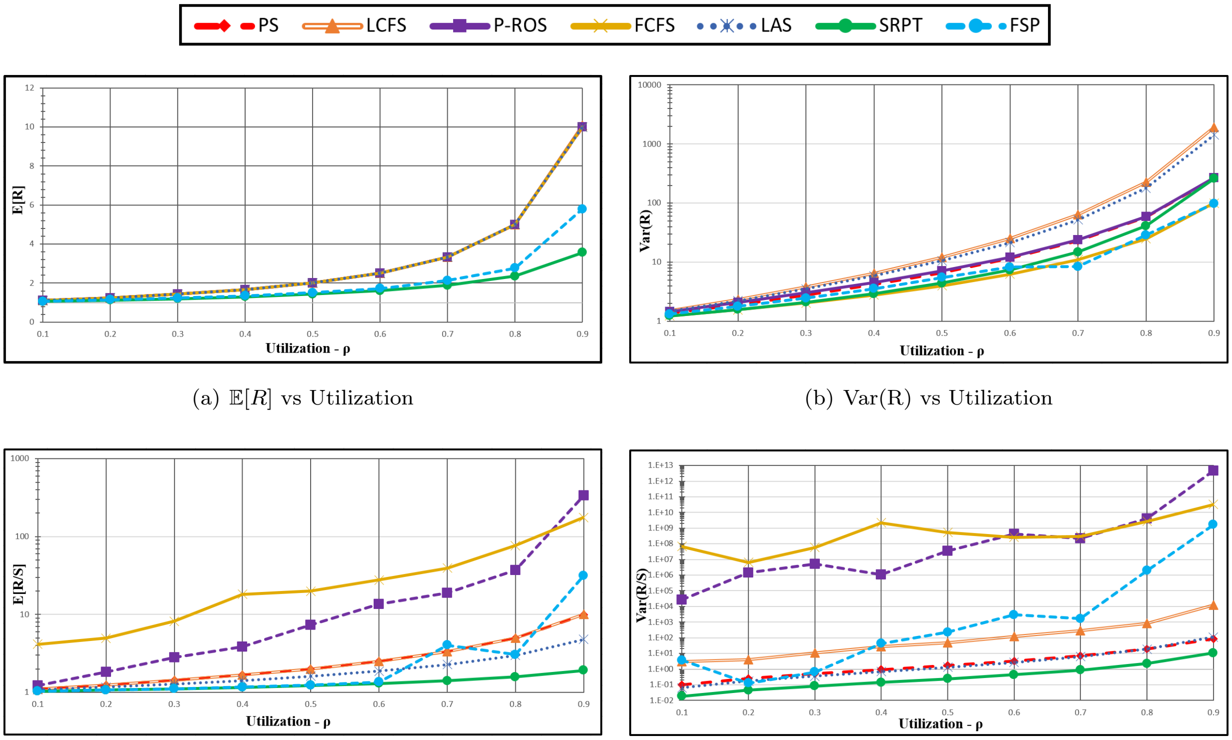
Until now, there was an important set of distributions which we excluded from our simulations. That is, distributions which have a significant probability of generating jobs which have an extremely small size. An example of such a distribution is the well known exponential distribution. Here service times are not bounded away from zero, and moreover, the mode of the density equals zero (an indicator that small jobs are likely to be generated). As will be seen, such distributions present difficulties when simulating non-preemptive policies, when moments of the slowdown are a concern.

Looking to Figure 8, this effect can be viewed explicitly. Here the service times are exponentially distributed, i.e. the system is an $M/M/1$ queue. Viewing the FCFS and P-ROS curves for SDV, one finds them to be jagged. This is not a product of true steady state behaviour, but rather a consequence of the system converging slowly to said steady state behaviour. That is, the simulation results can potentially be dominated by a small number of jobs (or in this case, one). It is noted that this slow convergence is emphasized more in non-preemptive policies. Consider a system with a single job present with a remaining service time equal to the mean service time. Now consider the case when a job arrives which is very small compared to the mean service time. For the sake of discussion, assume the new job is one thousandth the size of the mean. Under a policy without preemption that new job will have a slowdown of at least one thousand since it has to wait for the current job to complete. This is an inordinately large slowdown, and in turn impacts the SDV drastically. However, because the probability of such a small job being generated, while non-negligible, is still small, simulation results can be skewed to such a job being generated (or not generated) during its runtime.

To demonstrate this slow convergence is indeed due to extremely small jobs, we direct the reader to Figure 9. In this experiment, service times follow an exponential distribution shifted slightly to the right, i.e. a lower bound is imposed on the service times. When this is done one notes the curves exhibit an anticipated shape. That is, in contrast to the simulation with exponentially distributed service times, the curves are smooth and increasing with the load. This effect can be seen further (and more drastically) when comparing Figure 10 (hyper-exponential service times) to Figure 6 (hyper-exponential service times with a lower bound), where again, the hyper-exponential distribution has a significant probability of generating extremely small jobs.

Having a lower bound on the service times is not an unreasonable assumption in many domains, e.g. computing, healthcare, telecommunications, etc. As such, we argue that in practice these systems would exhibit a reasonable convergence to steady state, unlike that exhibited by the curves in Figures 8 and 10. In other words, the slow convergence for service time distributions which generate extremely small jobs is more of a theoretical issue, rather than a practical one.

Observation 5. *When the service time distribution is prone to generating arbitrarily small jobs, non-preemptive policies converge slowly to steady state. This in turn makes simulating the SDV for such systems problematic.*



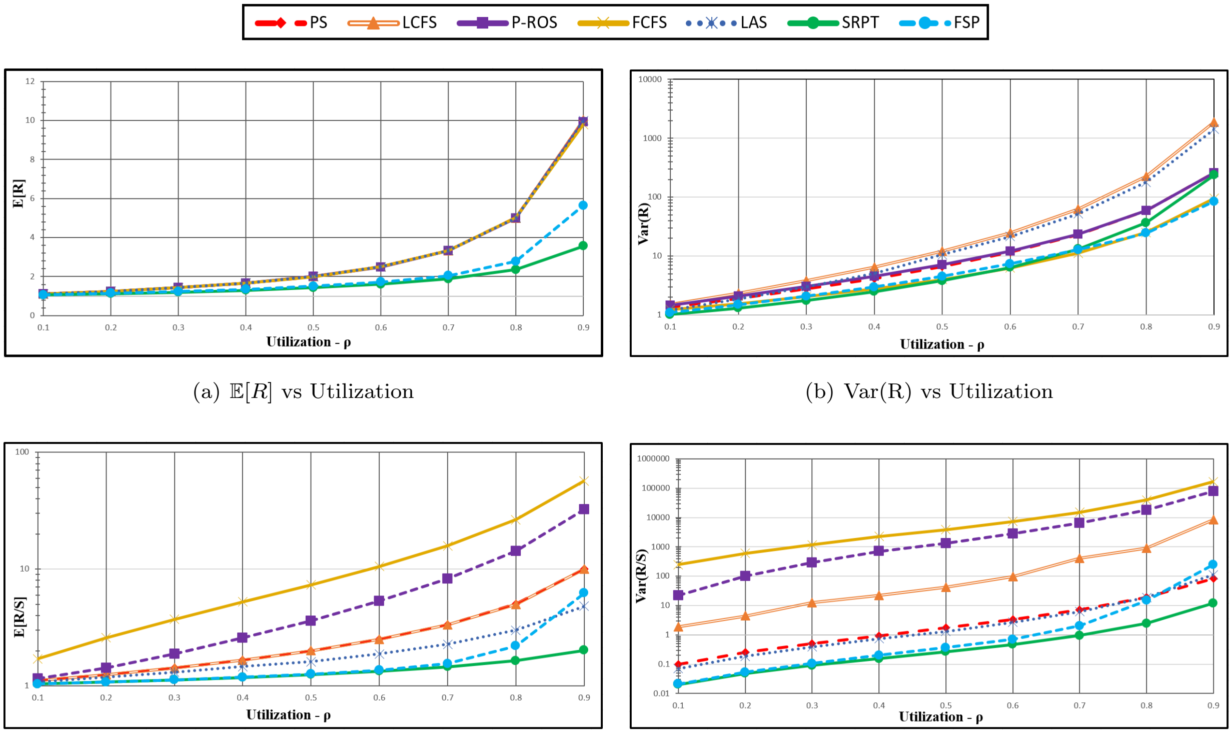
(a) $\mathbb{E}[R]$ vs Utilization

(b) Var(R) vs Utilization

(c) $\mathbb{E}[R/S]$ vs Utilization

(d) Var(R/S) vs Utilization

Figure 8: Simulation results for systems where service times are exponentially distributed: $1/\mu = 1$.



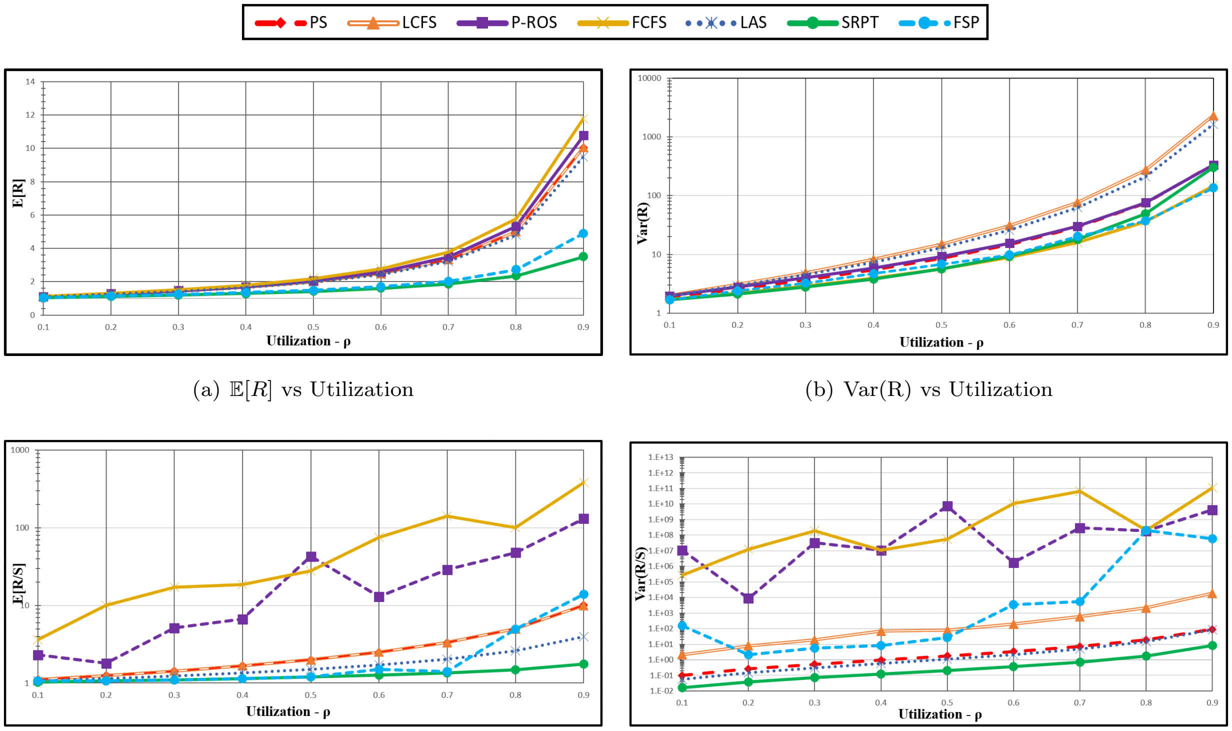
(a) $\mathbb{E}[R]$ vs Utilization

(b) Var(R) vs Utilization

(c) $\mathbb{E}[R/S]$ vs Utilization

(d) Var(R/S) vs Utilization

Figure 9: Simulation results for systems where service times are exponentially distributed but bounded away from zero: $1/\mu = 0.99$, and a lower bound of 0.01 (giving a mean of 1).



(a) $\mathbb{E}[R]$ vs Utilization

(b) Var(R) vs Utilization

(c) $\mathbb{E}[R/S]$ vs Utilization

(d) Var(R/S) vs Utilization

Figure 10: Numerical results for systems where service times follow a hyper-exponential distribution: $p_1 = 0.75$, $p_2 = 0.25$, $1/\mu_1 = 1.25$, $1/\mu_2 = 0.25$, where p_i denotes the probability of sampling from an exponential distribution with corresponding rate μ_i

3.5 Summary

Although it was seen that determining if one policy is more fair than another can be sensitive to the service time distribution, some general observations can be made which hold across all of our simulations. Firstly, and perhaps not surprisingly, the SDV of all policies increases with the load. There is a good deal of intuition behind this. One notes that for a system with a load close to zero, the vast majority of jobs will arrive to an empty system and are likely be processed before another job arrives. Thus most jobs will have a slowdown of one, and therefore most jobs will have equal slowdowns; implying the system SDV to be close to zero. On the other hand, when the load is close to one, the variance of the number of jobs that an arriving job sees is high, and furthermore, the variance of the number of jobs that arrive before said job is processed is also high. This in turn leads to a high variance in response times, and high variance in slowdown.

Observation 6. *Var*(R/S) *appears to increase with utilization.*

Measures of slowdown are often thought of as a mix between fairness and performance. One of the primary motivations to developing a fairness metric different from the expected slowdown was the bothersome equivalence, $\mathbb{E}[R/S]^{\text{PS}} = \mathbb{E}[R/S]^{\text{LCFS}}$ [14], since taken alongside the well known equivalence $\mathbb{E}[R]^{\text{PS}} = \mathbb{E}[R]^{\text{LCFS}}$, would imply that LCFS and PS are equally fair. This goes against a strong intuition (as we argued previously), that simply by construction PS is more fair than LCFS. When examining our simulations however, one notes that for all simulated distributions LCFS has a higher SDV than PS, and is therefore less fair. In other words, PS always performs just as well as LCFS, while also remaining the more fair of the two policies.

Observation 7. *For all experiments, Var*$(R/S)^{PS} <$ *Var*$(R/S)^{LCFS}$ *and from analysis it is known* $\mathbb{E}[R]^{PS} = \mathbb{E}[R]^{LCFS}$.

Continuing with PS, it is noted that while for many of the simulations PS is a competitive policy with respect to fairness, this is not always the case. This is perhaps surprising since it would seem PS is fair by construction (at every point in time every job has an equal slice of the processor). Specifically, for distributions with low variance, PS can be seen as an unfair policy. Thus, enforcing fairness at every point in time, can lead to an unfair system overall. Another interesting result from these simulations regarding PS is the load dependency of FSP. While previously observed that the ordering of SDV is load dependent, FSP seems to be the most sensitive to it. Since FSP works off of PS it is clear that all jobs do just as well under FSP as they would under PS, and on the surface this may seem like a favourable result (and it is for performance), but as discussed in Section 2, such a preference to small jobs can result in unfair treatment to larger ones, and in turn create an unfair system (often times putting FSP in the lower half of the simulated policies for higher loads).

Observation 8. *While PS is thought of as fair by construction, according to Var(R/S) it can in fact lead to unfair systems overall, in particular when the service time distribution has low variance.*

Although for Pareto distributed service times SRPT was shown to be quite unfair due to starvation, under almost all other simulated distributions, SRPT was shown to be the most fair policy. This is a favourable observation because it is well known that SRPT stochastically minimizes the response time of the $M/G/1$ queue. As noted before, it is possible for a policy to win on all fronts, i.e. have the lowest expected response time as well as the lowest SDV. It seems with regard to SRPT this observation can be made in a stronger fashion.

Observation 9. *For all experiments, for all simulated policies π, $\mathbb{E}[R]^{SRPT} \leq \mathbb{E}[R]^{\pi}$ and Var$(R/S)^{SRPT} \leq$ Var$(R/S)^{\pi}$ except under Uniform and Pareto distributions.*

These results complement works and surveys that have been done on SRPT. An investigation into the unfairness of SRPT is given in [6, 20] where the authors comment that while some jobs of a certain size may be treated very unfairly, SRPT may offer better slowdown in expectation in the majority of cases. In [20] SRPT was classified as "Sometimes Fair" and the authors suggest that SRPT can be used in a wide variety of situations. Even if the conditional mean slowdown is high for some distributions it might not impact how fair a policy is as a whole. For distributions with high variance it is intuitive that the SDV of SRPT would be high. When examining Pareto service times, one can informally view jobs as two classes: small jobs and large jobs, where the large jobs are several orders of magnitude greater than the mean. Then the system can be approximated as one with strict static priorities between the two classes (small jobs have priority over large jobs). Once this is realized the issue of starvation and unfairness becomes even clearer.

4 Conclusion

While fairness is a subjective concept, we identified several shortcomings of existing slowdown-based metrics. As such, we made a case for using the slowdown variance (SDV) as a metric for fairness, where the lower a system's SDV, the more fair it is. Through simulation we showed several key insights into the behaviour of SDV which, across a wide range of service time distributions, overcame the aforementioned shortcomings of previous metrics. Specifically, for the $M/G/1$ queue, SDV always considers PS to be more fair than LCFS, SDV considers SRPT to be extremely unfair under Pareto distributed service times, due to starvation and giving an unfair advantage to small jobs, and SDV provides a decoupling of performance from fairness. Regarding future work on the subject, we would like to formally show some of the observations made through simulation, such as arrive at closed form expressions for $\mathrm{Var}(R/S)^{\mathrm{PS}}$, $\mathrm{Var}(R/S)^{\mathrm{LCFS}}$, and $\mathrm{Var}(R/S)^{\mathrm{SRPT}}$ for general or specific service time distributions; as an alternative to closed form expressions tight bounds on the SDV may also be insightful. Furthermore, more numerical experiments can be run on systems other than the $M/G/1$ queue, such as $M/G/C$, open Jackson networks, etc. Clearly, SDV behaviours have yet to be explored in great detail, but we hope to start a discussion surrounding it as research on fairness evolves. We also hope that some of the insights here inform the general discussion of what it means for a queueing system to be fair.

Acknowledgement: This research was funded by the Natural Sciences and Engineering Research Council of Canada. We would also like to thank the two reviewers of this paper for their constructive feedback as well as their valuable guidance through past research.

References

[1] Source code. http://www.cas.mcmaster.ca/~macciov/publications.html. Accessed: 2017-04-12.

[2] B. Avi-Itzhak, E. Brosh, and H. Levy. SQF: A slowdown queueing fairness measure. *Performance Evaluation*, 64(12):1121–1136, 2007.

[3] B. Avi-Itzhak and H. Levy. On measuring fairness in queues. *Advances in Applied Probability*, 36(3):919–936, 2004.

[4] B. Avi-Itzhak, H. Levy, and D. Raz. A resource allocation queueing fairness measure: properties and bounds. *Queueing Systems*, 56(2):65–71, 2007.

[5] B. Avi-itzhak, H. Levy, and D. Raz. Quantifying fairness in queuing systems: principles, approaches, and applicability. *Probability in the Engineering and Informational Sciences*, 22(4):495–517, 2008.

[6] N. Bansal and M. Harchol-Balter. Analysis of SRPT scheduling: Investigating unfairness. In *Proceedings of the 2001 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, pages 279–290, 2001.

[7] H. Casanova, F. Desprez, and F. Suter. On cluster resource allocation for multiple parallel task graphs. *Journal of Parallel and Distributed Computing*, 70(12):1193–1203, 2010.

[8] H. Feng and V. Misra. Mixed scheduling disciplines for network flows. *SIGMETRICS Performance Evaluation Review*, 31(2):36–39, 2003.

[9] E. J. Friedman and S. G. Henderson. Fairness and efficiency in web server protocols. *SIGMETRICS Performance Evaluation Review*, 31(1):229–237, 2003.

[10] M. Harchol-Balter. *Performance modeling and design of computer systems: Queueing theory in action.* Cambridge University Press, 2013.

[11] R. Krashinsky and H. Balakrishnan. Minimizing energy for wireless web access with bounded slowdown. *Wireless Networks*, 11(2):135–148, 2005.

[12] A. J. Oliner, R. K. Sahoo, J. E. Moreira, M. Gupta, and A. Sivasubramaniam. Fault-aware job scheduling for BlueGene/L systems. In *Proceedings of the 18th International Symposium on Parallel and Distributed Processing*, pages 64–74. IEEE Computer Society, 1999.

[13] D. Raz, H. Levy, and B. Avi-Itzhak. A resource-allocation queueing fairness measure. *SIGMETRICS Performamce Evaluation Review*, 32(1):130–141, 2004.

[14] R. Righter, J. G. Shanthikumar, and G. Yamazaki. On extremal service disciplines in single-stage queueing systems. *Journal of Applied Probability*, 27(2):409–416, 1990.

[15] W. Sandmann. A discrimination frequency based queueing fairness measure with regard to job seniority and service requirement. In *Proceedings of Euro-NGI Conference of Next Generation Internet Networks*, pages 106–113, 2005.

[16] W. Sandmann. Quantitative fairness for assessing perceived service quality in queues. *Operational Research*, 13(2):153–186, 2013.

[17] A. Wierman. Fairness and classifications. *SIGMETRICS Performance Evaluation Review*, 34(4):4–12, 2007.

[18] A. Wierman. *Scheduling for todays computer systems: Bridging theory and practice.* PhD thesis, Carnegie Mellon University, 2007.

[19] A. Wierman. Fairness and scheduling in single server queues. *Surveys in Operations Research and Management Science*, 16(1):39–48, 2011.

[20] A. Wierman and M. Harchol-Balter. Classifying scheduling policies with respect to unfairness in an $M/GI/1$. In *Proceedings of the 2003 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, SIGMETRICS '03, pages 238–249, 2003.

[21] D. N. Zotkin and P. J. Keleher. Job-Length estimation and performance in backfilling schedulers. In *Proceedings of the Eighth International Symposium on High Performance Parallel and Distributed Computing*, pages 236–243. IEEE Computer Society, 1999.