Simple Near-Optimal Scheduling for the M/G/1

ZIV SCULLY, Carnegie Mellon University, USA
MOR HARCHOL-BALTER, Carnegie Mellon University, USA
ALAN SCHELLER-WOLF, Carnegie Mellon University, USA

We consider the problem of preemptively scheduling jobs to minimize mean response time of an M/G/1 queue. When we know each job's size, the *shortest remaining processing time* (SRPT) policy is optimal. Unfortunately, in many settings we do not have access to each job's size. Instead, we know only the job size distribution. In this setting the *Gittins* policy is known to minimize mean response time, but its complex priority structure can be computationally intractable. A much simpler alternative to Gittins is the *shortest expected remaining processing time* (SERPT) policy. While SERPT is a natural extension of SRPT to unknown job sizes, it is unknown whether or not SERPT is close to optimal for mean response time.

We present a new variant of SERPT called *monotonic SERPT* (M-SERPT) which is as simple as SERPT but has provably near-optimal mean response time at all loads for any job size distribution. Specifically, we prove the mean response time ratio between M-SERPT and Gittins is at most 3 for load $\rho \leq 8/9$ and at most 5 for any load. This makes M-SERPT the only non-Gittins scheduling policy known to have a constant-factor approximation ratio for mean response time.

CCS Concepts: • **General and reference** → **Performance**; • **Mathematics of computing** → **Queueing theory**; • **Networks** → **Network performance modeling**; • **Theory of computation** → *Routing and network design problems*; • **Computing methodologies** → *Model development and analysis*; • **Software and its engineering** → *Scheduling*.

Additional Key Words and Phrases: M/G/1; response time; latency; sojourn time; Gittins policy; shortest expected remaining processing time (SERPT), monotonic SERPT (M-SERPT); approximation ratio; multilevel processor sharing (MLPS); foreground-background (FB); shortest remaining processing time (SRPT)

ACM Reference Format:
Ziv Scully, Mor Harchol-Balter, and Alan Scheller-Wolf. 2020. Simple Near-Optimal Scheduling for the M/G/1. *Proc. ACM Meas. Anal. Comput. Syst.* 4, 1, Article 11 (March 2020), 29 pages. https://doi.org/10.1145/3379477

1 INTRODUCTION

Scheduling to minimize mean response time in a preemptive M/G/1 queue is a classic problem in queueing theory. When job sizes are known, the *shortest remaining processing time* (SRPT) policy is known to minimize mean response time [27]. Unfortunately, determining or estimating a job's exact size is difficult or impossible in many applications, in which case SRPT is impossible to implement. In such cases we only learn jobs' sizes after they have completed, which can give us a good estimate of the *distribution* of job sizes.

Authors' addresses: Ziv Scully, Carnegie Mellon University, Computer Science Department, 5000 Forbes Ave, Pittsburgh, PA, 15213, USA, zscully@cs.cmu.edu; Mor Harchol-Balter, Carnegie Mellon University, Computer Science Department, 5000 Forbes Ave, Pittsburgh, PA, 15213, USA, harchol@cs.cmu.edu; Alan Scheller-Wolf, Carnegie Mellon University, Tepper School of Business, 5000 Forbes Ave, Pittsburgh, PA, 15213, USA, awolf@andrew.cmu.edu.

2476-1249/2020/3-ART11 $15.00
https://doi.org/10.1145/3379477

When individual job sizes are unknown but the job size distribution is known, the *Gittins* policy minimizes mean response time [5, 13]. Gittins has a seemingly simple structure:

- Based on the job size distribution, Gittins defines a *rank function* that maps a job's *age*, which is the amount of service it has received so far, to a *rank*, which denotes its priority [28].
- At every moment in time, Gittins applies the rank function to each job's age and serves the job with the best rank.

Unfortunately, hidden in this simple outline is a major obstacle: computing the rank function from the job size distribution requires solving a nonconvex optimization problem for every possible age. Although the optimization can be simplified for specific classes of job size distributions [5], it is intractable in general.

In light of the difficulty of computing the Gittins rank function, practitioners turn to a wide variety of simpler scheduling policies, each of which has good performance in certain settings. Three of the most famous are the following:

- *First-come, first-serve* (FCFS) serves jobs nonpreemptively in the order they arrive.
  - FCFS generally performs well for low-variance job size distributions and is optimal for those with the *new better than used in expectation* property [5, 26].
- *Foreground-background* (FB) always serves the job of minimal age, splitting the server evenly in case of ties.
  - FB generally performs well for high-variance job size distributions and is optimal for those with the *decreasing hazard rate* property [5, 12, 25, 26].
- *Processor sharing* (PS) splits the server evenly between all jobs currently in the system.
  - PS has appealing insensitivity [9, 11, 18] and fairness [24, 29] properties which ensure passable mean response time for all job size distributions, but it is only optimal in the trivial special case of exponential job size distributions.

These are a few of the many scheduling heuristics studied in the past several decades [1, 4, 14, 15, 19, 23, 30, 31]. Unfortunately, there are *no guarantees of near-optimal mean response time* for any non-Gittins policy that hold across all job size distributions. In fact, we show in Appendix A that FCFS, FB, and PS can have *infinite* mean response time ratio compared to Gittins. We therefore ask:

> Is there a *simple* scheduling policy with near-optimal mean response time for all job size distributions?

One candidate for such a policy is *shortest expected remaining processing time* (SERPT). Like Gittins, SERPT assigns each job a rank as a function of its age, but SERPT has a much simpler rank function: a job's rank is its *expected remaining size*. That is, if the job size distribution is X, then under SERPT, a job's rank at age a is

$$r_{\mathsf{SERPT}}(a) = \mathbf{E}[X - a \mid X > a],$$

where lower rank means better priority. Intuitively, it seems like SERPT should have low mean response time because it prioritizes jobs that are short in expectation, analogous to what SRPT does for known job sizes. SERPT is certainly much simpler than Gittins, as summarized in Table 1.1 and discussed in detail in Appendix B.

- For discrete job size distributions with n support points, the best known algorithms compute Gittins's rank function in $O(n^2)$ time [10]. In contrast, SERPT's rank function takes just $O(n)$ time to compute.
- For continuous job size distributions, computing Gittins's rank function is intractable with known methods: it requires solving a nonconvex optimization problem at every age a, and the objective of the optimization requires numerical integration to compute. In contrast, SERPT's rank function requires just numerical integration.

Table 1.1. Comparison of Gittins, SERPT, and M-SERPT

Policy	Computation		Optimality
	Discrete	*Continuous*	
Gittins	$O(n^2)$	intractable	optimal
SERPT	$O(n)$	tractable	unknown
M-SERPT	$O(n)$	tractable	5-approximation or better

1.1 Challenges

SERPT is intuitively appealing and simple to compute, but does it have near-optimal mean response time? This question is open: there is *no known bound* on the performance gap between SERPT and Gittins. To be precise, letting[1]

$$C_{\text{SERPT}}(X) = \frac{\mathbf{E}[T_{\text{SERPT}}(X)]}{\mathbf{E}[T_{\text{Gittins}}(X)]}$$

be the mean response time ratio between SERPT and Gittins for a given job size distribution X, there is no known bound on

$$\text{approximation ratio of SERPT} = \sup_X C_{\text{SERPT}}(X).$$

This approximation ratio is difficult to bound because we have to consider *all* possible job size distributions X.

In fact, until recently it was unknown how to compute $C_{\text{SERPT}}(X)$ even given a *specific* job size distribution X. This changed with the introduction of the *SOAP* technique [28], which can analyze the mean response time of any scheduling policy that can be specified by a rank function. We can use SOAP to *numerically* compute $C_{\text{SERPT}}(X)$ for any given job size distribution X. However, SOAP does not give a bound on SERPT's approximation ratio, which requires considering all possible X.

One might hope to derive a general expression for $C_{\text{SERPT}}(X)$ using SOAP. While this is possible in principle, the resulting expression is intractable (Section 3.2). In light of this, our strategy is to create a new scheduling policy that captures the essence of SERPT but has a tractable mean response time expression in terms of X.

1.2 A New Simple Scheduling Policy: M-SERPT

In this paper we introduce a new policy called *monotonic SERPT* (M-SERPT) that is simple to compute and has provably near-optimal mean response time. Like Gittins and SERPT, we specify M-SERPT using a rank function. M-SERPT's rank function is like SERPT's, except *a job's rank never improves:*

$$r_{\text{M-SERPT}}(a) = \max_{0 \le b \le a} r_{\text{SERPT}}(b).$$

We prove that M-SERPT is a 5-approximation for mean response time, meaning its mean response time is at most 5 times that of Gittins. This makes M-SERPT the first non-Gittins scheduling policy known to have a constant-factor approximation ratio. The approximation ratio is even smaller at low and moderate loads. For example, M-SERPT is a 3-approximation for load $\rho \le 8/9$. Remarkably, M-SERPT achieves its constant-factor approximation ratio with a rank function that is as simple to compute as SERPT's (Table 1.1).

Our approximation ratio for M-SERPT is a worst-case upper bound. There are many distributions where M-SERPT's performance is equal or very close to Gittins's. For example, Fig. 1.1 compares the

[1]The mean response time ratio $C_{\text{SERPT}}(X)$ also depends on the load ρ, but we omit ρ from the notation to reduce clutter.

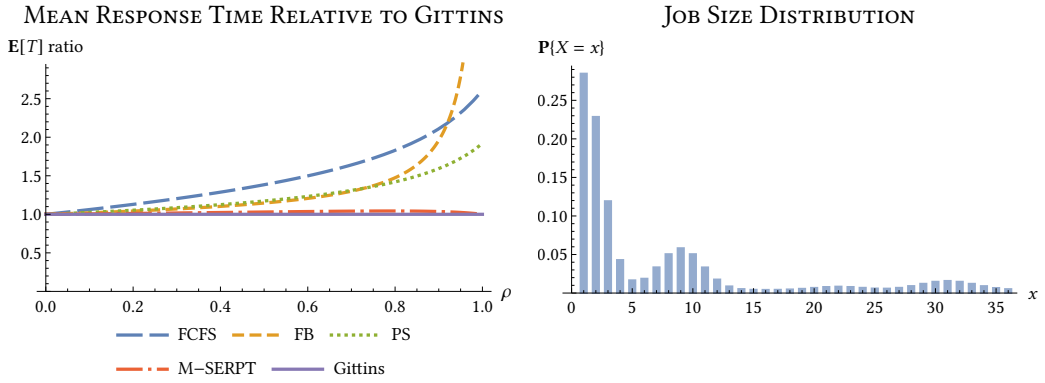

Fig. 1.1. Mean Response Time Comparison

mean response times of several policies, including M-SERPT, to that of Gittins, where the job size distribution is the mixture of four bell curves pictured. In this example, M-SERPT's mean response time is within 4% of Gittins's across all loads.[2] In further preliminary numerical experiments, omitted for lack of space, we only observed a mean response time difference of more than 15% in a specific pathological scenario (Section 7).

1.3 Contributions

We introduce M-SERPT, the first non-Gittins policy proven to achieve mean response time within a constant factor of Gittins's. Our specific contributions are as follows:

- We define the *monotonic SERPT* (M-SERPT) policy, a new variant of SERPT (Section 2).
- We introduce a new simplification of the SOAP response time analysis that yields a tractable mean response time expression for M-SERPT (Sections 3 and 4).
- We prove that M-SERPT is a 5-approximation for minimizing mean response time, with an even smaller approximation ratio at low and moderate loads (Section 5).
- We use the fact that M-SERPT is a 5-approximation to resolve two open questions in M/G/1 scheduling theory (Section 6).
- We construct a pathological job size distribution for which the mean response time ratio between M-SERPT and Gittins is 2, which is the largest ratio we have observed (Section 7).

M-SERPT's approximation ratio is therefore between 2 and 5. We conclude by discussing in detail why this gap is hard to close and pointing out several possible avenues of attack (Section 8).

1.4 Related Work

In this paper we consider minimizing mean response time in the setting of an M/G/1 queue with unknown job sizes but known job size distribution. We are not aware of prior work on approximation ratios in this exact setting, but there is prior work in related settings.

Wierman et al. [30] study the M/G/1 with *known* job sizes. They prove that all scheduling policies in a class called *SMART* are 2-approximations for mean response time, where the baseline for this setting is SRPT [27]. All SMART policies use job size information, so they cannot be applied to our setting of unknown job sizes. Proving approximation ratios in our setting is significantly more

[2]For the specific distribution in Fig. 1.1, SERPT has mean response time between Gittins and M-SERPT. However, there are examples where SERPT has greater mean response time than M-SERPT, and whether SERPT has a constant-factor approximation ratio remains an open problem.

challenging because the scheduling policies involved, namely M-SERPT and Gittins, have much more complicated mean response time formulas than SRPT and the SMART class [28, 30].

We now turn to settings with unknown job sizes. Kalyanasundaram and Pruhs [16] propose a policy called *randomized multilevel feedback* (RMLF) for the case where neither job sizes *nor the job size distribution* are known. RMLF has been studied in two specific settings:

- In the *worst-case* setting, meaning job sizes and arrival times are chosen adversarially, RMLF has mean response time $O(\log n)$ times that of SRPT, where n is the number of jobs in the arrival sequence [8, 16]. Up to constant factors, this is the best possible performance in the worst-case setting [22].
- In the *stochastic* GI/GI/1 setting, Bansal et al. [7] prove that as the load ρ approaches 1,

$$\frac{\mathbf{E}[T_{\text{RMLF}}]}{\mathbf{E}[T_{\text{SRPT}}]} = O\left(\log \frac{1}{1-\rho}\right).$$

These results differ from ours in two important ways. First, the results do not prove constant-factor approximation ratios: they give asymptotic ratios that become arbitrarily large in the $n \to \infty$ and $\rho \to 1$ limits, respectively. In contrast, we show that M-SERPT is a 5-approximation at all loads ρ, even in the $\rho \to 1$ limit. Second, the results compare RMLF with SRPT, not with Gittins, even though job sizes are unknown. This is because optimal policies for the worst-case and GI/GI/1 settings are not known, especially with unknown job size distribution, leaving SRPT as a sensible baseline for comparison. In contrast, in the M/G/1 setting with known job size distribution, we know the optimal policy is Gittins, so we compare M-SERPT to Gittins. Comparing RMLF to Gittins is an interesting open problem.

A final setting is a hybrid between the worst-case and M/G/1 settings. Megow and Vredeveld [21] consider scheduling jobs with stochastic sizes but adversarially chosen arrival times. However, rather than considering the metric of mean response time, they consider mean *completion* time. The difference between these metrics is that a job's response time is measured relative to its arrival, whereas a job's completion time is measured relative to time 0. Completion and response times are only the same when all the jobs arrive at once. Thus, while Megow and Vredeveld [21] show that Gittins and a related policy are 2-approximations for mean completion time, this does not translate into an approximation ratio for mean response time.

2 SYSTEM MODEL AND PRELIMINARIES

We consider scheduling policies for a single-class M/G/1 queue in which jobs have unknown size. We write λ for the arrival rate and X for the job size distribution, so the load is $\rho = \lambda \mathbf{E}[X]$. We assume $\rho < 1$ for stability. Jobs may be preempted at any time without delay or loss of work.

Throughout this paper, all monotonicities are meant in the weak sense unless otherwise specified. For example, "increasing" means "nondecreasing". Many quantities defined in this paper depend on one or both of X and ρ, but we usually leave this implicit in our notation to reduce clutter.

We write $\overline{F}$ and f for the tail and density functions of X, respectively. For ease of presentation, we assume that

- f is well defined and continuous, implying the distribution does not have atoms; and
- both the SERPT rank function (Definition 2.2) and the hazard rate function

$$h(x) = \frac{f(x)}{\overline{F}(x)}$$

  are piecewise monotonic, ruling out some pathological cases.

With some effort, one very likely can adapt our proofs to relax these assumptions. In particular, we have confirmed our results for discrete job size distributions, omitting the details for lack of space.

We write T_π for the response time distribution under policy π, and we write $T_\pi(x)$ for the response time distribution of a job of size x under policy π. We use similar notation for waiting time Q_π and residence time R_π (Section 3.1) For the most part, π is one of

- G, denoting Gittins;
- S, denoting SERPT; or
- MS, denoting M-SERPT.

These policies are defined in Section 2.1. We use the same subscripts for other quantities that depend on the scheduling policy. We omit the subscript when discussing a generic SOAP policy.

2.1 SOAP Policies and Rank Functions

A SOAP policy [28] is specified by a *rank function*

$$r : \mathbb{R}_{\geq 0} \to \mathbb{R}$$

which maps a job's *age*, the amount of time it has been served, to its *rank*, or priority.[3] All SOAP policies have the same core scheduling rule: always serve the job of *minimum rank*, breaking ties in first-come, first served (FCFS) order.

Gittins, SERPT, and M-SERPT are all SOAP policies. Their rank functions are defined as follows.

Definition 2.1. The *Gittins* policy is the SOAP policy with rank function

$$r_{\mathsf{G}}(a) = \inf_{b>a} \frac{\int_a^b \overline{F}(t)\,\mathrm{d}t}{\overline{F}(a) - \overline{F}(b)}.$$

Definition 2.2. The *shortest expected remaining processing time* (SERPT) policy is the SOAP policy with rank function

$$r_{\mathsf{S}}(a) = \mathbf{E}[X - a \mid X > a] = \frac{\int_a^\infty \overline{F}(t)\,\mathrm{d}t}{\overline{F}(a)}.$$

Definition 2.3. The *increasing envelope* of function r is

$$r^\uparrow(a) = \max_{0 \leq b \leq a} r(b).$$

Definition 2.4. The *monotonic SERPT* (M-SERPT) policy is the SOAP policy whose rank function is the increasing envelope of SERPT's rank function:

$$r_{\mathsf{MS}}(a) = r_{\mathsf{S}}^\uparrow(a) = \max_{0 \leq b \leq a} r_{\mathsf{S}}(b).$$

Figure 2.1 illustrates an example of the relationship between the SERPT and M-SERPT rank functions. Under our assumptions on the job size distribution, each of Gittins, SERPT, and M-SERPT has a continuous, piecewise monotonic rank function [6].

3 KEY IDEAS

We now give a high-level overview of how we prove our main result, namely an upper bound on M-SERPT's approximation ratio. The purpose of this section is to communicate, with minimal notation, (1) the main ideas of our proof and (2) the novelty of our approach. As such, we discuss simplified versions of our key definitions and lemmas, deferring the full versions to later in the paper. For example, our main result in Theorem 5.1 bounds M-SERPT's approximation ratio as a function of load, but here we focus on a simpler corollary:

$$\mathbf{E}[T_{\mathsf{MS}}] \leq 5\mathbf{E}[T_{\mathsf{G}}]. \tag{3.1}$$

[3] The full SOAP definition [28] allows a job's rank to also depend on characteristics such as its size or class, but we do not need this generality for the policies in this paper.

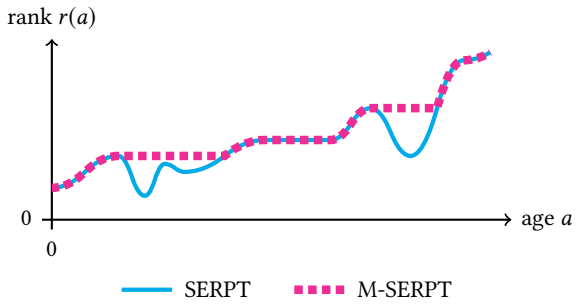

Fig. 2.1. Example of SERPT and M-SERPT Rank Functions

3.1 Waiting Time and Residence Time

To prove (3.1), we first split response time into two pieces:

- *residence time* R, which is the response time of jobs that arrive to an empty system; and
- *waiting time* Q, which is the extra delay due to the fact that the system is not always empty.

For SOAP policies, response time is equal in distribution to the independent sum of the waiting and residence times [28]:

$$T = Q + R.$$

The bound in (3.1) follows from two main lemmas, one bounding each of M-SERPT's mean waiting and residence times. Specifically, Lemma 5.6 implies

$$\mathbf{E}[Q_{\text{MS}}] \leq 2\mathbf{E}[Q_{\text{G}}], \tag{3.2}$$

and Lemma 5.7 implies

$$\mathbf{E}[R_{\text{MS}}] \leq \mathbf{E}[Q_{\text{MS}}] + \mathbf{E}[T_{\text{G}}]. \tag{3.3}$$

The proofs of (3.2) and (3.3) constitute the main technical contribution of our work, as their combination immediately yields (3.1):

$$\begin{aligned}\mathbf{E}[T_{\text{MS}}] &= \mathbf{E}[Q_{\text{MS}}] + \mathbf{E}[R_{\text{MS}}] \\ &\leq 2\mathbf{E}[Q_{\text{MS}}] + \mathbf{E}[T_{\text{G}}] \\ &\leq 4\mathbf{E}[Q_{\text{G}}] + \mathbf{E}[T_{\text{G}}] \\ &\leq 5\mathbf{E}[T_{\text{G}}].\end{aligned}$$

3.2 Why SOAP Is Not Enough

How might we prove (3.2) and (3.3)? One might think of using the SOAP response time analysis of Scully et al. [28]. Their main result [28, Theorem 5.5] takes a rank function r and yields closed-form expressions for $\mathbf{E}[Q]$ and $\mathbf{E}[R]$. By "closed-form" expressions, we mean functions of the job size distribution's tail function $\overline{F}$ and the load ρ that can be written with just arithmetic and integrals. However, the dependence on r is much more complicated. This is a major obstacle for M-SERPT and Gittins because their rank functions depend on the job size distribution. This makes it intractable to directly apply the SOAP analysis to comparing M-SERPT with Gittins over all job size distributions.

Much of the complexity of the SOAP analysis of Scully et al. [28] comes from being general enough to handle multiclass systems, namely those in which different jobs follow different rank functions. We only consider single-class systems in this paper. Our approach is therefore to simplify the SOAP analysis to our single-class setting (Section 3.3). This results in much simpler expressions for $\mathbf{E}[Q]$

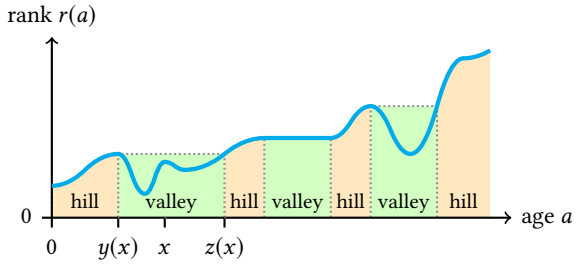

Fig. 3.1. Hills and Valleys

and $\mathbf{E}[R]$, partly because we are willing to settle for bounds. The resulting simple expressions make it possible to compare M-SERPT to Gittins over all job size distributions (Sections 3.4 and 3.5).

3.3 Hills and Valleys

Suppose we are using a SOAP policy with rank function r. We use $r^\uparrow$, the increasing envelope of r (Definition 2.3), to classify ages into two types:

- *hill ages*, those at which $r^\uparrow$ is strictly increasing; and
- *valley ages*, those at which $r^\uparrow$ is constant.

We call an interval of hill ages or valley ages a *hill* or *valley*, respectively. Figure 3.1, which shows an example of hills and valleys, clarifies two points:

- Hill ages are those at which $r^\uparrow$, not just r, is strictly increasing. For a to be a hill age, not only must r be increasing at age a, but r must not attain a greater rank at any earlier age.
- Valley ages are those at which $r^\uparrow$, not r, is constant. In general, r might increase, decrease, or be constant at valley ages.

Given a size x, we define two ages:

- the *previous hill age* $y(x)$ is the greatest hill age $\le x$, and
- the *next hill age* $z(x)$ is the least hill age $\ge x$.

If x is a hill age, then $y(x) = x = z(x)$, and if x is a valley age, then $y(x) < x < z(x)$, as illustrated in Fig. 3.1.[4]

For any SOAP policy, we can bound $\mathbf{E}[Q]$ and $\mathbf{E}[R]$ in terms of y and z. Proposition 4.7 implies

$$\mathbf{E}[Q] \ge \int_0^\infty \frac{\tau(z(x))}{\bar{\rho}(y(x)) \cdot \bar{\rho}(z(x))} f(x)\,\mathrm{d}x, \tag{3.4}$$

and Proposition 4.8 implies

$$\mathbf{E}[R] \le \int_0^\infty \frac{x}{\bar{\rho}(y(x))} f(x)\,\mathrm{d}x, \tag{3.5}$$

with both bounds becoming equalities for M-SERPT. Here $\bar{\rho}$ and τ (Definitions 4.5 and 4.6) are functions that do not depend on the scheduling policy.

Hills and valleys are important for two reasons. First, the expressions in (3.4) and (3.5) depend on the scheduling policy only via $y(x)$ and $z(x)$, the previous and next hill ages of each size x. This means relating the mean response times of M-SERPT and Gittins partly reduces to relating the hills and valleys of M-SERPT and Gittins. Second, as we will soon see, hills and valleys turn out to be important tools for organizing the computations in the proofs of our two main bounds, (3.2) and (3.3).

[4]We address some corner cases in the definitions of hills, valleys, y, and z in Section 4.1.

3.4 Outline of Waiting Time Bound

We now outline the proof of (3.2), namely $\mathbf{E}[Q_{\mathrm{MS}}] \le 2\mathbf{E}[Q_{\mathrm{G}}]$. By (3.4),

$$\frac{\mathbf{E}[Q_{\mathrm{MS}}]}{\mathbf{E}[Q_{\mathrm{G}}]} \le \frac{\displaystyle\int_0^\infty \frac{\tau(z_{\mathrm{MS}}(x))}{\bar\rho(y_{\mathrm{MS}}(x)) \cdot \bar\rho(z_{\mathrm{MS}}(x))} f(x)\,\mathrm{d}x}{\displaystyle\int_0^\infty \frac{\tau(z_{\mathrm{G}}(x))}{\bar\rho(y_{\mathrm{G}}(x)) \cdot \bar\rho(z_{\mathrm{G}}(x))} f(x)\,\mathrm{d}x}. \tag{3.6}$$

Our strategy for proving (3.2) is to split the integration regions in (3.6) into chunks and prove the bound for each chunk $[u, v]$:

$$\frac{\displaystyle\int_u^v \frac{\tau(z_{\mathrm{MS}}(x))}{\bar\rho(y_{\mathrm{MS}}(x)) \cdot \bar\rho(z_{\mathrm{MS}}(x))} f(x)\,\mathrm{d}x}{\displaystyle\int_u^v \frac{\tau(z_{\mathrm{G}}(x))}{\bar\rho(y_{\mathrm{G}}(x)) \cdot \bar\rho(z_{\mathrm{G}}(x))} f(x)\,\mathrm{d}x} \le 2. \tag{3.7}$$

The key to this approach is to choose the right chunks. It turns out that a good choice is for each Gittins hill and valley to be a chunk.

As mentioned at the end of Section 3.3, a key to comparing M-SERPT to Gittins is comparing their hills and valleys. We show in Lemma 5.3 that every Gittins hill age is also an M-SERPT hill age, but not necessarily vice versa. This implies that for any size x,

$$y_{\mathrm{G}}(x) \le y_{\mathrm{MS}}(x) \le x \le z_{\mathrm{MS}}(x) \le z_{\mathrm{G}}(x). \tag{3.8}$$

Proving (3.7) when chunk $[u, v]$ is a Gittins hill case is simple. When x is a Gittins hill age, (3.8) collapses to an equality, so the left-hand side of (3.7) is 1.

Proving (3.7) when chunk $[u, v]$ is a Gittins valley is much more complicated. As illustrated in Fig. 3.1, for all $x \in [u, v]$, we have $y_{\mathrm{G}}(x) = u$ and $z_{\mathrm{G}}(x) = v$, simplifying the denominator in (3.7). Since τ is increasing (Table 5.1), it suffices to show $q(u, v) \le 2$, where

$$q(a, b) = \int_a^b \frac{\bar\rho(u) \cdot \bar\rho(v)}{\bar\rho(y_{\mathrm{MS}}(x)) \cdot \bar\rho(z_{\mathrm{MS}}(x))} \cdot \frac{f(x)}{\bar F(u) - \bar F(v)}\,\mathrm{d}x. \tag{3.9}$$

We bound $q(u, v)$ by splitting it into $q(u, v) = q(u, x_*) + q(x_*, v)$ for some x_*. Because the $\bar\rho$ ratio in the integrand is increasing in x (Table 5.1), the idea is to carefully choose x_* such that

- $q(u, x_*) \le 1$ because the $\bar\rho$ ratio is not too large for $x \in [u, x_*]$, and
- $q(x_*, v) \le 1$ because, roughly speaking, $f(x)$ is not too large for $x \in [x_*, v]$.

It turns out there is a natural choice for x_*, and the above strategy works when x_* is an M-SERPT hill age. When x_* is an M-SERPT valley age, we have to split $q(u, v)$ into three pieces instead, with the third piece handling the valley containing x_*, but the upper bounds on the three pieces still add up to at most 2.

The proof of Lemma 5.6 in Section 5.2 closely follows the strategy outlined in this section. The main difference between (3.2) and Lemma 5.6 is that the latter's bound is smaller at lower load.

3.5 Outline of Residence Time Bound

We now outline the proof of (3.3), namely $\mathbf{E}[R_{\mathrm{MS}}] \le \mathbf{E}[Q_{\mathrm{MS}}] + \mathbf{E}[T_{\mathrm{G}}]$. The first and more important step is Lemma 5.7, which says

$$\mathbf{E}[R_{\mathrm{MS}}] \le \mathbf{E}[Q_{\mathrm{MS}}] + \left(\frac{1}{\rho} \log \frac{1}{1-\rho}\right)\mathbf{E}[X]. \tag{3.10}$$

The second step uses a result of Wierman et al. [30, Theorem 5.8] to upper bound the last term in (3.10) by $\mathbf{E}[T_{\mathrm{G}}]$, which yields (3.3).

Our strategy for proving (3.10) is, roughly speaking, to integrate (3.4) and (3.5) by parts:

$$\mathbf{E}[Q_{\mathrm{MS}}] = \int_0^\infty \bar{F}(x) \cdot \text{“} \frac{\mathrm{d}}{\mathrm{d}x} \frac{\tau(z_{\mathrm{MS}}(x))}{\bar{\rho}(y_{\mathrm{MS}}(x)) \cdot \bar{\rho}(z_{\mathrm{MS}}(x))} \text{”} \, \mathrm{d}x$$
$$\mathbf{E}[R_{\mathrm{MS}}] = \int_0^\infty \bar{F}(x) \cdot \text{“} \frac{\mathrm{d}}{\mathrm{d}x} \frac{x}{\bar{\rho}(y_{\mathrm{MS}}(x))} \text{”} \, \mathrm{d}x.$$

These integral expressions are not rigorous and are presented for intuition only. Specifically, y_{MS} and z_{MS} have discontinuities, so the derivatives are not well defined everywhere, thus the quotation marks. Again we split the integrals into chunks, this time based on M-SERPT hills and valleys, and prove the bound for each chunk.

Proving the bound for M-SERPT hills is simple because $y_{\mathrm{MS}}(x) = x = z_{\mathrm{MS}}(x)$ when x is an M-SERPT hill age. This means the derivatives are well defined, and they even have a term in common, making them easy to compare. In fact, we do not need any special properties of M-SERPT for this part of the argument.

Proving the bound for M-SERPT valleys is more complicated. Discontinuities of y_{MS} and z_{MS} occur at the boundaries of valleys. Handling this requires some care, but we nevertheless obtain simple expressions for the waiting time and residence time chunks. The main difficulty is that the expressions are difficult to compare. It is this comparison that requires special properties of M-SERPT.

The proof of Lemma 5.7 in Section 5.3 closely follows the strategy outlined in this section. However, when we put everything together to prove our main result, Theorem 5.1, it turns out jumping from (3.10) to (3.3) is only a good idea at very high loads, whereas using (3.10) directly yields a better bound at most loads.

4 HILLS AND VALLEYS

Hills and valleys are new concepts that play several important roles in our bound of M-SERPT's approximation ratio (Sections 3.3–3.5). The purpose of this section is to formally state definitions and results relating to hills and valleys. Throughout this section we work with a generic SOAP policy with rank function r.

4.1 Defining Hills and Valleys

Definition 4.1.

- A *valley age* is an age $a > 0$ at which the increasing envelope $r^\uparrow$ of the rank function r is locally constant, meaning there exists some $\varepsilon > 0$ such that $r^\uparrow(b) = r^\uparrow(a)$ for all $b \in (a - \varepsilon, a + \varepsilon)$.
- A *hill age* is an age that is not a valley age.

Definition 4.2.

- The *previous hill age* of size x is the latest hill age before x:

$$y(x) = \sup\{a < x \mid a \text{ is a hill age}\}.$$

- The *next hill age* of size x is the earliest hill age after x:[5]

$$z(x) = \inf\{a \geq x \mid a \text{ is a hill age}\}.$$

The difference in inequality strictness between y and z comes from how y and z are used to bound mean waiting and residence times (Appendix C). For the most part, $y(x) = x = z(x)$ for any hill age x, but there is an exception when x is preceded by an interval $(x - \varepsilon, x)$ of valley ages. This distinction is occasionally important, so we extend our terminology to capture it.

[5]There is a corner case for $x = 0$: we define $y(0) = 0$ and $z(0) = z(0+)$, where postfix + denotes a right limit.

Definition 4.3.
- A *hill size* is a size x such that $y(x) = x = z(x)$.
- A *valley size* is a size that is not a hill size.

Definition 4.4.
- A *hill* is an interval of hill sizes.
- A *valley* is an interval $(u, v]$ of valley sizes where u and v are hill ages.

Definitions 4.2 and 4.4 are illustrated in Fig. 3.1. The distinction between hill ages and hill sizes is important only for the upper boundaries of valleys, which are hill ages but not hill sizes.[6]

4.2 Response Time Bounds

We now use hills and valleys to write down simple bounds on $\mathbf{E}[Q(x)]$ and $\mathbf{E}[R(x)]$, the expected waiting and residence times (Section 3.1), respectively, of a job of size x.

Definition 4.5. The *a-truncated load complement* is one minus what the load of the system would be if every job's size were truncated at age a:

$$\bar{\rho}(a) = 1 - \lambda \mathbf{E}[\min\{X, a\}] = 1 - \int_0^a \lambda \bar{F}(t)\,\mathrm{d}t.$$

Definition 4.6. The *a-truncated second moment factor* is

$$\tau(a) = \frac{\lambda}{2}\mathbf{E}[(\min\{X, a\})^2] = \int_0^x \lambda t \bar{F}(t)\,\mathrm{d}t.$$

Proposition 4.7. *Under any SOAP policy, the expected waiting time of a job of size x is bounded by*

$$\mathbf{E}[Q(x)] \geq \frac{\tau(z(x))}{\bar{\rho}(y(x)) \cdot \bar{\rho}(z(x))},$$

with equality if the policy has a monotonic rank function.

Proposition 4.8. *Under any SOAP policy, the expected residence time of a job of size x is bounded by*

$$\mathbf{E}[R(x)] \leq \frac{x}{\bar{\rho}(y(x))},$$

with equality if the policy has a monotonic rank function.

Proofs of Propositions 4.7 and 4.8. See Appendix C.

M-SERPT has a monotonic rank function, so both Propositions 4.7 and 4.8 yield useful equalities for M-SERPT. However, to prove an upper bound on M-SERPT's approximation ratio, we want lower bounds for Gittins, for which only Proposition 4.7 is useful. Instead of using Proposition 4.8 for Gittins, we use the following lower bounds.

Proposition 4.9. *Under any SOAP policy, the mean residence time is bounded by* $\mathbf{E}[R] \geq \mathbf{E}[X]$.

Proof. A job's residence time is, by definition (Section 3.1), at least its size. □

Proposition 4.10. *Under any scheduling policy, the mean response time is bounded by*

$$\mathbf{E}[T] \geq \left(\frac{1}{\rho}\log\frac{1}{1-\rho}\right)\mathbf{E}[X].$$

Proof. Wierman et al. [30, Theorem 5.8] show that the desired lower bound holds for SRPT, which has lower mean response time than any other policy [27]. □

[6] There is another corner case for 0: it is always a hill age, but it is not a hill size if $z(0+) > 0$.

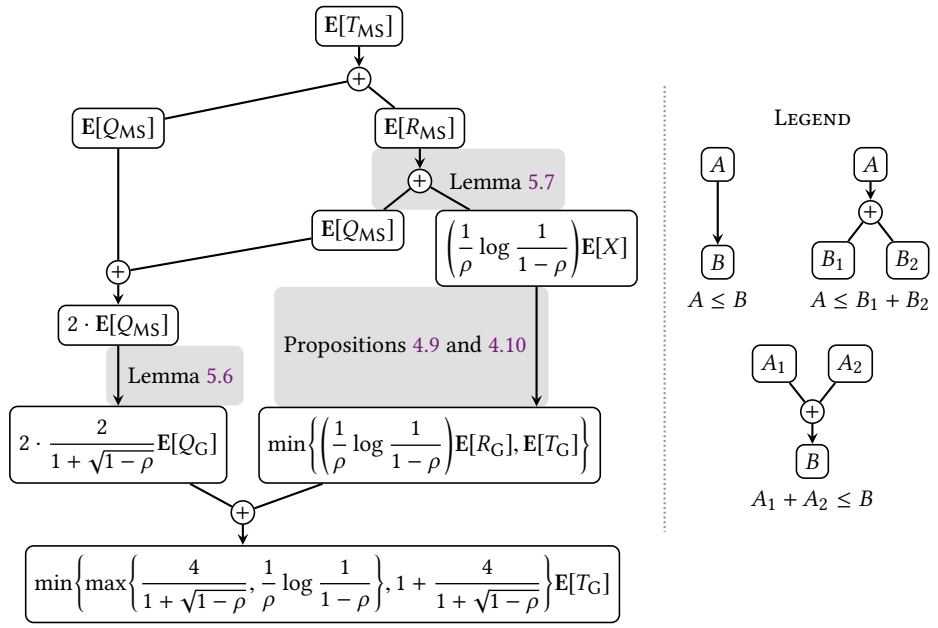

Fig. 5.1. Bounding Mean Response Time of M-SERPT

5 UPPER BOUND ON M-SERPT'S APPROXIMATION RATIO

In this section we prove our main result, which is an upper bound on the mean response time ratio between M-SERPT and Gittins.

Theorem 5.1. *The mean response time ratio between M-SERPT and Gittins is bounded by*[7]

$$\frac{\mathbf{E}[T_{MS}]}{\mathbf{E}[T_G]} \le \begin{cases} \dfrac{4}{1+\sqrt{1-\rho}} & 0 \le \rho < 0.9587 \\ \dfrac{1}{\rho}\log\dfrac{1}{1-\rho} & 0.9587 \le \rho < 0.9898 \\ 1+\dfrac{4}{1+\sqrt{1-\rho}} & 0.9898 \le \rho < 1. \end{cases}$$

Proof. See Fig. 5.1 and Appendix D.

As illustrated in Fig. 5.1, the main steps in the proof of Theorem 5.1 are Lemmas 5.6 and 5.7 (Sections 5.2 and 5.3). Figure 5.2 plots the resulting bound as a function of load ρ. The following corollary gives intuition for this function in terms of concrete values.

Corollary 5.2. *For the problem of preemptive scheduling to minimize mean response time in an M/G/1 queue with unknown job sizes, the approximation ratio of M-SERPT is at most*

- 2.5 *for load* $\rho \le 0.64$,
- 3 *for load* $\rho \le 8/9 \approx 0.89$,
- 3.3 *for load* $\rho \le 0.95$,
- 4 *for load* $\rho \le 0.98$, *and*
- 5 *for all loads.*

[7]The numbers 0.9587 and 0.9898 are approximations accurate to 4 decimal places.

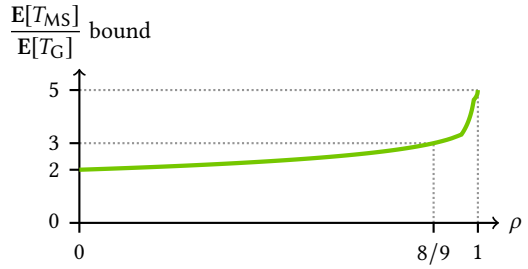

Fig. 5.2. Bound on Mean Response Time Ratio

5.1 Properties of M-SERPT Hill Ages

In this section we prove some properties of M-SERPT hills and valleys, and in particular M-SERPT hill ages. We begin by relating the hills and valleys of M-SERPT and Gittins. The following lemma builds on ideas introduced by Aalto et al. [6], but it is a novel result.[8]

Lemma 5.3. *Every Gittins hill age is also an M-SERPT hill age, and similarly for hill sizes.*

Proof. See Appendix D.

We now show a key property of M-SERPT hill ages that lets us to bound $\bar{\rho}$ ratios, such as those in (3.9), in terms of $\bar{F}$ ratios.

Lemma 5.4. *For any M-SERPT hill age b and any $a \le b$,*

$$\frac{\bar{\rho}(a)}{\bar{\rho}(b)} \le \frac{1}{1-\rho+\rho\frac{\bar{F}(b)}{\bar{F}(a)}}.$$

Proof. Recall from Definition 2.4 that r_{MS} is the increasing envelope of r_{S}. By Definition 4.3, this means M-SERPT has the same hill and valley ages as SERPT. We therefore have

- $r_{\mathrm{S}}(a) \le r_{\mathrm{MS}}(a)$ by Definition 2.4,
- $r_{\mathrm{MS}}(a) \le r_{\mathrm{MS}}(b)$ because r_{MS} is increasing, and
- $r_{\mathrm{S}}(b) = r_{\mathrm{MS}}(b)$ because b is a SERPT hill age.

Putting these together gives us $r_{\mathrm{S}}(a) \le r_{\mathrm{S}}(b)$, which by Definition 2.2 is the same as

$$\frac{\int_a^\infty \bar{F}(t)\,\mathrm{d}t}{\bar{F}(a)} \le \frac{\int_b^\infty \bar{F}(t)\,\mathrm{d}t}{\bar{F}(b)}.$$

Multiplying both sides by λ and applying Definition 4.5 yields

$$\frac{\bar{\rho}(a)-(1-\rho)}{\bar{F}(a)} \le \frac{\bar{\rho}(b)-(1-\rho)}{\bar{F}(b)}.$$

Letting $\zeta = \bar{F}(b)/\bar{F}(a)$, this rearranges to

$$\frac{\bar{\rho}(b)}{\bar{\rho}(a)} \ge \zeta + (1-\zeta)\frac{1-\rho}{\bar{\rho}(a)}.$$

Because $\bar{\rho}(a) \le 1$, the right-hand side is at least $1-\rho+\zeta\rho$, which implies the desired inequality. □

[8] In particular, Lemma 5.3 is not equivalent to Proposition 7 of Aalto et al. [6] because hills are not simply the ages at which the rank function is increasing (Definition 4.4).

Table 5.1. Monotonicity Facts

Function	Monotonicity	Defined in
$\overline{F}$	decreasing	Section 2
$\bar{\rho}$	decreasing	Definition 4.5
τ	increasing	Definition 4.6
y, z	increasing	Definition 4.2

The bound in Lemma 5.4 is increasing in ρ, implying the following simpler bound.

Corollary 5.5. *For any M-SERPT hill age b and any $a \le b$,*

$$\frac{\bar{\rho}(a)}{\bar{\rho}(b)} \le \frac{\overline{F}(a)}{\overline{F}(b)}.$$

5.2 Waiting Time Bound

The proofs in the remainder of this section frequently use the monotonicity facts listed in Table 5.1. As a reminder, all monotonicities are meant in the weak sense unless otherwise specified. For example, "decreasing" means nonincreasing. So as not to disrupt the flow of the proofs, we use facts from Table 5.1 with only a reference to the table.

Lemma 5.6. *The mean waiting time of M-SERPT is bounded by*

$$\frac{\mathbf{E}[Q_{MS}]}{\mathbf{E}[Q_G]} \le \frac{2}{1+\sqrt{1-\rho}}.$$

Proof. By Lemma 5.3, because $y_G(x) = x = z_G(x)$ for all Gittins hill sizes x, we have

$$\frac{\mathbf{E}[Q_{MS}(X) \mid X \text{ is a Gittins hill size}]}{\mathbf{E}[Q_G(X) \mid X \text{ is a Gittins hill size}]} \le 1.$$

Therefore, it suffices to show that for any Gittins valley $(u, v]$,

$$\frac{\mathbf{E}[Q_{MS}(X) \mid X \in (u,v]]}{\mathbf{E}[Q_G(X) \mid X \in (u,v]]} \le \frac{2}{1+\sqrt{1-\rho}}.$$

For any $x \in (u, v]$, Lemma 5.3 implies the following key fact:

$$u = y_G(x) \le y_{MS}(x) \le x \le z_{MS}(x) \le z_G(x) = v. \tag{5.1}$$

Applying Proposition 4.7 and Table 5.1, we obtain

$$\begin{aligned}\frac{\mathbf{E}[Q_{MS}(X) \mid X \in (u,v]]}{\mathbf{E}[Q_G(X) \mid X \in (u,v]]} &= \frac{\displaystyle\int_u^v \frac{\tau(z_{MS}(x))}{\bar{\rho}(y_{MS}(x)) \cdot \bar{\rho}(z_{MS}(x))} \cdot \frac{f(x)}{\overline{F}(u) - \overline{F}(v)} \, dx}{\displaystyle\frac{\tau(v)}{\bar{\rho}(u) \cdot \bar{\rho}(v)}} \\ &\le \int_u^v \frac{\bar{\rho}(u) \cdot \bar{\rho}(v)}{\bar{\rho}(y_{MS}(x)) \cdot \bar{\rho}(z_{MS}(x))} \cdot \frac{f(x)}{\overline{F}(u) - \overline{F}(v)} \, dx.\end{aligned}$$

Let

$$q(a,b) = \int_a^b \frac{\bar{\rho}(u) \cdot \bar{\rho}(v)}{\bar{\rho}(y_{MS}(x)) \cdot \bar{\rho}(z_{MS}(x))} \cdot \frac{f(x)}{\overline{F}(u) - \overline{F}(v)} \, dx.$$

It suffices to bound $q(u, v)$. To do so, we split the integration region into three pieces at carefully chosen ages y_* and z_*, then we bound each of $q(u, y_*)$, $q(y_*, z_*)$, and $q(z_*, v)$.

Before specifying y_* and z_*, we need two other definitions. First, for all $x \in (u, v]$, let

$$\bar{G}(x) = 1 - \rho + \rho\frac{\bar{F}(x)}{\bar{F}(u)}.$$

With this notation, Lemma 5.4 says that if x is an M-SERPT hill age, then[9]

$$\frac{\bar{\rho}(u)}{\bar{\rho}(x)} \leq \frac{\bar{G}(u)}{\bar{G}(x)}. \tag{5.2}$$

Second, let $x_* \in (u, v]$ be the age such that

$$\frac{\bar{G}(u)}{\bar{G}(x_*)} = \frac{\bar{\rho}(u)}{\bar{\rho}(v)}. \tag{5.3}$$

Such an age must exist by continuity of $\bar{G}$ because by Table 5.1 and (5.2),

$$\bar{G}(u) \geq \frac{\bar{\rho}(v)}{\bar{\rho}(u)}\bar{G}(u) \geq \bar{G}(v).$$

We can now define

$$y_* = y_{\mathrm{MS}}(x_*)$$
$$z_* = z_{\mathrm{MS}}(x_*).$$

We bound each of $q(u, y_*)$, $q(y_*, z_*)$, and $q(z_*, v)$ in Steps 1–3 below. The core of each step is bounding the ratios $\bar{\rho}(u)/\bar{\rho}(y_{\mathrm{MS}}(x))$ and $\bar{\rho}(v)/\bar{\rho}(z_{\mathrm{MS}}(x))$.

- By Table 5.1 and (5.1) we have

$$\frac{\bar{\rho}(v)}{\bar{\rho}(z_{\mathrm{MS}}(x))} \leq 1 \tag{5.4}$$

  and, using (5.3),

$$\frac{\bar{\rho}(u)}{\bar{\rho}(y_{\mathrm{MS}}(x))} = \frac{\bar{\rho}(v)}{\bar{\rho}(y_{\mathrm{MS}}(x))} \cdot \frac{\bar{G}(u)}{\bar{G}(x_*)} \leq \frac{\bar{G}(u)}{\bar{G}(x_*)}. \tag{5.5}$$

- Since $y_{\mathrm{MS}}(x)$ and $z_{\mathrm{MS}}(x)$ are M-SERPT hill ages, by (5.2) we have

$$\frac{\bar{\rho}(u)}{\bar{\rho}(y_{\mathrm{MS}}(x))} \leq \frac{\bar{G}(u)}{\bar{G}(y_{\mathrm{MS}}(x))} \tag{5.6}$$

  and, using (5.3),

$$\frac{\bar{\rho}(v)}{\bar{\rho}(z_{\mathrm{MS}}(x))} = \frac{\bar{\rho}(u)}{\bar{\rho}(z_{\mathrm{MS}}(x))} \cdot \frac{\bar{G}(x_*)}{\bar{G}(u)} \leq \frac{\bar{G}(x_*)}{\bar{G}(z_{\mathrm{MS}}(x))}. \tag{5.7}$$

In each of Steps 1–3, we apply either (5.4) or (5.7), whichever gives a tighter bound, and similarly for (5.5) and (5.6).

We need one last definition before carrying out Steps 1–3: to avoid mixing $\bar{F}$ and $\bar{G}$, let

$$g(x) = -\frac{\mathrm{d}}{\mathrm{d}x}\bar{G}(x) = \rho f(x),$$

which allows us to write

$$\frac{f(x)}{\bar{F}(u) - \bar{F}(v)} = \frac{g(x)}{\bar{G}(u) - \bar{G}(v)}.$$

[9]Even though $\bar{G}(u) = 1$, we find that explicitly writing $\bar{G}(u)$ in ratios with other uses of $\bar{G}$ makes the proof easier to follow.

Step 1: bounding $q(u, y_*)$. Since both u and y_* are M-SERPT hill ages, we can partition $(u, y_*]$ into M-SERPT hills and valleys,[10] meaning there exist

$$u = z_0 \le y_1 < z_1 < \ldots < y_n < z_n \le y_{n+1} = y_*$$

such that

- $(y_i, z_i]$ is an M-SERPT valley for all $i \in \{1, \ldots, n\}$,
- $(z_i, y_{i+1}]$ is an M-SERPT hill for all $i \in \{1, \ldots, n\}$, and
- either $z_0 = y_1$ or $(z_0, y_1]$ is an M-SERPT hill.

For each M-SERPT valley, we have $y_{\mathrm{MS}}(x) = y_i$ and $z_{\mathrm{MS}}(x) = z_i$ for $x \in (y_i, z_i]$, so applying (5.6) and (5.7) yields

$$\begin{aligned} q(y_i, z_i) &\le \int_{y_i}^{z_i} \frac{\bar{G}(u) \cdot \bar{G}(x_*)}{\bar{G}(y_i) \cdot \bar{G}(z_i)} \cdot \frac{g(x)}{\bar{G}(u) - \bar{G}(v)} \, \mathrm{d}x \\ &= \frac{\bar{G}(u) \cdot \bar{G}(x_*)}{\bar{G}(u) - \bar{G}(v)} \left(\frac{1}{\bar{G}(z_i)} - \frac{1}{\bar{G}(y_i)} \right). \end{aligned} \tag{5.8}$$

For each M-SERPT hill, we have $y_{\mathrm{MS}}(x) = x = z_{\mathrm{MS}}(x)$ for $x \in (z_i, y_{i+1}]$, so applying (5.6) and (5.7) yields

$$\begin{aligned} q(z_i, y_{i+1}) &\le \int_{z_i}^{y_{i+1}} \frac{\bar{G}(u) \cdot \bar{G}(x_*)}{\bar{G}(x)^2} \cdot \frac{g(x)}{\bar{G}(u) - \bar{G}(v)} \, \mathrm{d}x \\ &= \frac{\bar{G}(u) \cdot \bar{G}(x_*)}{\bar{G}(u) - \bar{G}(v)} \left(\frac{1}{\bar{G}(y_{i+1})} - \frac{1}{\bar{G}(z_i)} \right). \end{aligned} \tag{5.9}$$

Combining (5.8) and (5.9) for each M-SERPT hill and valley implies

$$\begin{aligned} q(u, y_*) &= \sum_{i=1}^{n} q(y_i, z_i) + \sum_{i=0}^{n} q(z_i, y_{i+1}) \\ &\le \frac{\bar{G}(u) \cdot \bar{G}(x_*)}{\bar{G}(u) - \bar{G}(v)} \left(\frac{1}{\bar{G}(y_{n+1})} - \frac{1}{\bar{G}(z_0)} \right) \\ &= \frac{\bar{G}(u)}{\bar{G}(u) - \bar{G}(v)} \left(\frac{\bar{G}(x_*)}{\bar{G}(y_*)} - \frac{\bar{G}(x_*)}{\bar{G}(u)} \right). \end{aligned}$$

Step 2: bounding $q(y_*, z_*)$. If x_* is an M-SERPT hill size, then $q(y_*, z_*) = q(x_*, x_*) = 0$. Otherwise, since $y_{\mathrm{MS}}(x) = y_*$ for all $x \in (y_*, z_*]$, applying (5.4) and (5.6) yields

$$\begin{aligned} q(y_*, z_*) &\le \int_{y_*}^{z_*} \frac{\bar{G}(u)}{\bar{G}(y_*)} \cdot \frac{g(x)}{\bar{G}(u) - \bar{G}(v)} \, \mathrm{d}x \\ &= \frac{\bar{G}(u)}{\bar{G}(u) - \bar{G}(v)} \left(1 - \frac{\bar{G}(z_*)}{\bar{G}(y_*)} \right). \end{aligned}$$

Step 3: bounding $q(z_*, v)$. Applying (5.4) and (5.5) yields

$$\begin{aligned} q(z_*, v) &\le \int_{z_*}^{v} \frac{\bar{G}(u)}{\bar{G}(x_*)} \cdot \frac{g(x)}{\bar{G}(u) - \bar{G}(v)} \, \mathrm{d}x \\ &= \frac{\bar{G}(u)}{\bar{G}(u) - \bar{G}(v)} \left(\frac{\bar{G}(z_*)}{\bar{G}(x_*)} - \frac{\bar{G}(v)}{\bar{G}(x_*)} \right). \end{aligned}$$

[10] The potential obstacle to partitioning is that u or y_* might be in the interior of a valley (Definition 4.4), but u and y_* being hill ages ensures this is not the case.

Combining the results of Steps 1–3 gives us

$$q(u,v) \le \frac{\bar{G}(u)}{\bar{G}(u)-\bar{G}(v)}\left(\frac{\bar{G}(x_*)}{\bar{G}(y_*)} - \frac{\bar{G}(x_*)}{\bar{G}(u)} + 1 - \frac{\bar{G}(z_*)}{\bar{G}(y_*)} + \frac{\bar{G}(z_*)}{\bar{G}(x_*)} - \frac{\bar{G}(v)}{\bar{G}(x_*)}\right). \tag{5.10}$$

Table 5.1 and (5.1) imply

$$\frac{\bar{G}(z_*)}{\bar{G}(x_*)} - \frac{\bar{G}(z_*)}{\bar{G}(y_*)} \le 1 - \frac{\bar{G}(x_*)}{\bar{G}(y_*)},$$

and minimizing over possible values of $\bar{G}(x_*)$ gives

$$\frac{\bar{G}(x_*)}{\bar{G}(u)} + \frac{\bar{G}(v)}{\bar{G}(x_*)} \ge 2\sqrt{\frac{\bar{G}(v)}{\bar{G}(u)}}.$$

Applying these to (5.10) and using the fact that $\bar{G}(v)/\bar{G}(u) \ge 1-\rho$ yields

$$\begin{aligned} q(u,v) &\le \frac{\bar{G}(u)}{\bar{G}(u)-\bar{G}(v)}\left(2 - 2\sqrt{\frac{\bar{G}(v)}{\bar{G}(u)}}\right) \\ &= \frac{2}{1+\sqrt{\frac{\bar{G}(v)}{\bar{G}(u)}}} \\ &\le \frac{2}{1+\sqrt{1-\rho}}. \end{aligned}$$

□

5.3 Residence Time Bound

Lemma 5.7. *The mean residence time of M-SERPT is bounded by*

$$\mathbf{E}[R_{MS}] \le \mathbf{E}[Q_{MS}] + \left(\frac{1}{\rho}\log\frac{1}{1-\rho}\right)\mathbf{E}[X].$$

Proof. We can partition $\mathbb{R}_{\ge 0}$ into M-SERPT hills and valleys, meaning there exist

$$0 = z_0 < y_1 < z_1 < \dots$$

such that

- $(y_i, z_i]$ is an M-SERPT valley for all $i \ge 1$,
- $(z_i, y_{i+1}]$ is an M-SERPT hill for all $i \ge 1$, and
- either $z_0 = y_0$ or $(z_0, y_1]$ is an M-SERPT hill.

Let

$$\begin{aligned} \Delta_Q(a,b) &= \mathbf{E}[Q_{\mathrm{MS}}(\min\{X,b\})] - \mathbf{E}[Q_{\mathrm{MS}}(\min\{X,a\})] \\ \Delta_R(a,b) &= \mathbf{E}[R_{\mathrm{MS}}(\min\{X,b\})] - \mathbf{E}[R_{\mathrm{MS}}(\min\{X,a\})] \\ \Delta_{\log}(a,b) &= \frac{1}{\lambda}\log\frac{1}{\bar{\rho}(b)} - \frac{1}{\lambda}\log\frac{1}{\bar{\rho}(a)}. \end{aligned}$$

We wish to show $\Delta_R(0,\infty) \le \Delta_Q(0,\infty) + \Delta_{\log}(0,\infty)$. It suffices to show that for each M-SERPT hill $(z_i, y_{i+1}]$,[11]

$$\Delta_R(z_i+, y_{i+1}-) \le \Delta_Q(z_i+, y_{i+1}-) + \Delta_{\log}(z_i, y_{i+1}), \tag{5.11}$$

and that for each M-SERPT valley $(y_i, z_i]$,

$$\Delta_R(y_i-, z_i+) \le \Delta_Q(y_i-, z_i+) + \Delta_{\log}(y_i, z_i). \tag{5.12}$$

[11] We use postfix − and + to denote left and right limits, respectively. They are not needed for $\Delta_{\log}$, which is continuous.

We prove these bounds in Steps 1 and 2 below, respectively. In both steps we use the fact that

$$\frac{\mathrm{d}}{\mathrm{d}x}\mathbf{E}[Q_{\mathrm{MS}}(\min\{X,x\})] = \bar{F}(x)\cdot\frac{\mathrm{d}}{\mathrm{d}x}\mathbf{E}[Q_{\mathrm{MS}}(x)]$$
$$\frac{\mathrm{d}}{\mathrm{d}x}\mathbf{E}[R_{\mathrm{MS}}(\min\{X,x\})] = \bar{F}(x)\cdot\frac{\mathrm{d}}{\mathrm{d}x}\mathbf{E}[R_{\mathrm{MS}}(x)].$$

Step 1: bound for M-SERPT hills. We have $y_{\mathrm{MS}}(x) = x = z_{\mathrm{MS}}(x)$ for all $x \in (z_i, y_{i+1})$. Recalling Definitions 4.5 and 4.6, by Proposition 4.7,

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}x}\Delta_Q(z_i+,x) &= \frac{\mathrm{d}}{\mathrm{d}x}\mathbf{E}[Q_{\mathrm{MS}}(\min\{X,x\})]\\ &= \bar{F}(x)\cdot\frac{\mathrm{d}}{\mathrm{d}x}\frac{\tau(x)}{\bar{\rho}(x)^2}\\ &= \frac{\lambda x\bar{F}(x)^2}{\bar{\rho}(x)^2} + \frac{2\lambda\bar{F}(x)^2\cdot\tau(x)}{\bar{\rho}(x)^3}.\end{aligned}$$

Similarly, by Proposition 4.8,

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}x}\Delta_R(z_i+,x) &= \frac{\mathrm{d}}{\mathrm{d}x}\mathbf{E}[R_{\mathrm{MS}}(\min\{X,x\})]\\ &= \bar{F}(x)\cdot\frac{\mathrm{d}}{\mathrm{d}x}\frac{x}{\bar{\rho}(x)}\\ &= \frac{\bar{F}(x)}{\bar{\rho}(x)} + \frac{\lambda x\bar{F}(x)^2}{\bar{\rho}(x)^2}.\end{aligned}$$

Finally, we have

$$\frac{\mathrm{d}}{\mathrm{d}x}\Delta_{\log}(z_i,x) = \frac{\bar{F}(x)}{\bar{\rho}(x)}.$$

Examining the three derivatives, we see

$$\frac{\mathrm{d}}{\mathrm{d}x}\Delta_R(z_i+,x) \le \frac{\mathrm{d}}{\mathrm{d}x}\Delta_Q(z_i+,x) + \frac{\mathrm{d}}{\mathrm{d}x}\Delta_{\log}(z_i,x),$$

which implies (5.11), as desired.

Step 2: bound for M-SERPT valleys. We have $y_{\mathrm{MS}}(x) = y_i$ and $z_{\mathrm{MS}}(x) = z_i$ for all $x \in (y_i, z_i)$, which means

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}x}\Delta_Q(y_i-,x) &= 0\\ \frac{\mathrm{d}}{\mathrm{d}x}\Delta_R(y_i-,x) &= \frac{\bar{F}(x)}{\bar{\rho}(y_i)}\\ \frac{\mathrm{d}}{\mathrm{d}x}\Delta_{\log}(y_i,x) &= \frac{\bar{F}(x)}{\bar{\rho}(x)}.\end{aligned}$$

However, we must still account for discontinuities at $x = y_i$ and $x = z_i$.

We first prove a lower bound on $\Delta_Q(y_i-,z_i+)$. We have

$$\begin{aligned}\Delta_Q(y_i-,z_i+) &= \Delta_Q(y_i-,y_i+) + \Delta_Q(z_i-,z_i+)\\ &= \bar{F}(y_i)\left(\frac{\tau(z_i)}{\bar{\rho}(y_i)\cdot\bar{\rho}(z_i)} - \frac{\tau(y_i)}{\bar{\rho}(y_i)^2}\right) + \bar{F}(z_i)\left(\frac{\tau(z_i)}{\bar{\rho}(z_i)^2} - \frac{\tau(z_i)}{\bar{\rho}(y_i)\cdot\bar{\rho}(z_i)}\right).\end{aligned} \tag{5.13}$$

Both terms in (5.13) are nonnegative by Table 5.1. Applying Corollary 5.5 with $a = y_i$ and $b = z_i$ to the first term and dropping the second term yields

$$\Delta_Q(y_i-, z_i+) \geq \overline{F}(z_i)\left(\frac{\tau(z_i)}{\overline{\rho}(z_i)^2} - \frac{\tau(y_i)}{\overline{\rho}(y_i)\cdot\overline{\rho}(z_i)}\right). \tag{5.14}$$

We now turn to $\Delta_R(y_i-, z_i+)$. We have

$$\begin{aligned}\Delta_R(y_i-, z_i+) &= \Delta_R(y_i-, y_i+) + \Delta_R(y_i+, z_i-) + \Delta_R(z_i-, z_i+)\\ &= 0 + \int_{y_i}^{z_i} \frac{\overline{F}(x)}{\overline{\rho}(y_i)}\,\mathrm{d}x + \overline{F}(z_i)\left(\frac{z_i}{\overline{\rho}(z_i)} - \frac{z_i}{\overline{\rho}(y_i)}\right)\\ &= \Delta_{\log}(y_i, z_i) + \overline{F}(z_i)\left(\frac{z_i}{\overline{\rho}(z_i)} - \frac{z_i}{\overline{\rho}(y_i)}\right) - \int_{y_i}^{z_i} \overline{F}(x)\left(\frac{1}{\overline{\rho}(x)} - \frac{1}{\overline{\rho}(y_i)}\right)\mathrm{d}x. \end{aligned}\tag{5.15}$$

Applying Corollary 5.5 with $a = x$ and $b = z_i$ to the last term of (5.15) yields

$$\Delta_R(y_i-, z_i+) \leq \Delta_{\log}(y_i, z_i) + \overline{F}(z_i)\left(\frac{z_i}{\overline{\rho}(z_i)} - \frac{z_i}{\overline{\rho}(y_i)}\right) - \int_{y_i}^{z_i} \overline{F}(z_i)\left(\frac{1}{\overline{\rho}(z_i)} - \frac{\overline{\rho}(x)}{\overline{\rho}(y_i)\cdot\overline{\rho}(z_i)}\right)\mathrm{d}x. \tag{5.16}$$

Using integration by parts one can compute

$$\int_{y_i}^{z_i} \overline{\rho}(x)\,\mathrm{d}x = z_i\overline{\rho}(z_i) - y_i\overline{\rho}(y_i) + \tau(z_i) - \tau(y_i).$$

Substituting this into (5.16) causes many terms to cancel, leaving

$$\Delta_R(y_i-, z_i+) \leq \Delta_{\log}(y_i, z_i) + \overline{F}(z_i)\frac{\tau(z_i) - \tau(y_i)}{\overline{\rho}(y_i)\cdot\overline{\rho}(z_i)}$$

which combined with Table 5.1 and (5.14) implies (5.12), as desired. □

6 ADDITIONAL IMPLICATIONS OF M-SERPT'S APPROXIMATION RATIO

In this section we discuss additional implications of the fact that M-SERPT is a constant-factor approximation of Gittins, resolving two open questions in M/G/1 scheduling theory. Section 6.1 addresses the performance of FB for job size distributions with the *increasing mean residual lifetime* (IMRL) property, and Section 6.2 addresses the performance achievable by policies in the *multilevel processor sharing* (MLPS) class.

6.1 Performance of FB for IMRL Job Size Distributions

Definition 6.1. A job size distribution X has the (strictly) *increasing mean residual lifetime* (IMRL) property if a job's expected remaining size $\mathbf{E}[X - a \mid X > a]$ is (strictly) increasing in its age a.

Consider the setting of an M/G/1 with an IMRL job size distribution. In this IMRL setting, the greater a job's age, the greater its expected remaining size. We therefore might expect that the FB policy, which prioritizes jobs of lower age, would yield low mean response time. In fact, it was believed for some time that FB was optimal for the IMRL setting [26]. However, Aalto and Ayesta [2] found a flaw in the proof, along with a counterexample IMRL job size distribution for which FB is not optimal. While Aalto and Ayesta [2] show that FB has lower mean response time than PS in the IMRL setting, whether FB is close to optimal for the IMRL setting is an open question.

The following corollary resolves this question for the case of strictly IMRL job size distributions. It turns out that M-SERPT and FB are equivalent in this case, because the strictly IMRL property implies M-SERPT's rank function is strictly increasing, just like FB's. This means FB has the same approximation ratio as M-SERPT for strictly IMRL job size distributions.

Corollary 6.2. *For the problem of preemptive scheduling to minimize mean response time in an M/G/1 queue with unknown job sizes, if the job size distribution is strictly IMRL, FB is a constant-factor approximation.*

6.2 Performance Achievable by MLPS Policies

Multilevel processor sharing (MLPS) policies are a class of preemptive scheduling policies introduced by Kleinrock [19]. An MLPS policy is specified by a list of threshold ages $0 = a_0, a_1, a_2, \ldots$, where interval $[a_i, a_{i+1}]$ is the *ith level*. Jobs with ages in lower levels have priority over those in higher levels, and within each level, jobs are scheduled using one of FCFS, FB, or PS. While we know how to analyze the mean response time of any MLPS policy [14, 19, 20], *optimizing* an MLPS policy, meaning choosing the threshold ages and scheduling policies within each level to minimize mean response time, is an open problem [3, 4].[12]

The following corollary takes a major step towards solving this problem. It turns out that M-SERPT is an MLPS policy: its levels are the hills and valleys, with FB used within each hill and FCFS used within each valley. While M-SERPT is not always the optimal MLPS policy, we know it performs within a constant factor of Gittins.

Corollary 6.3. *For any job size distribution, there exists an MLPS policy, namely M-SERPT, with mean response time a constant factor times that of Gittins.*

Combining this with results on the RMLF policy [7] implies the following additional corollary.[13]

Corollary 6.4. *For any job size distribution, there exists an MLPS policy, namely M-SERPT, whose mean response time ratio compared to SRPT is at most* $O(\log(1/(1-\rho)))$ *in the* $\rho \to 1$ *limit.*

7 LOWER BOUND ON M-SERPT'S APPROXIMATION RATIO

We have shown that M-SERPT is a 5-approximation for minimizing mean response time. The natural followup question is: what case is worst for M-SERPT? We have yet to find a scenario in which M-SERPT performs 5 times worse than Gittins. Instead, the largest ratio we have observed so far is 2. This occurs with the following pathological job size distribution, where $\delta \in (0, 1)$ is small:

$$X = \begin{cases} 1-\delta & \text{w.p. } 1-\delta \\ 1 & \text{w.p. } \delta - \delta^2 \\ \delta^{-1}+1 & \text{w.p. } \delta^2. \end{cases}$$

That is, nearly all jobs are size $1-\delta$, and nearly all the rest are size 1.

How do the M-SERPT and Gittins rank functions differ for X? Computing ranks using Definitions 2.1 and 2.4, we find

$$\begin{aligned} r_{\text{MS}}(0) &< r_{\text{MS}}(1-\delta) < r_{\text{MS}}(1) \\ r_{\text{G}}(1-\delta) &< \quad r_{\text{G}}(0) \quad < r_{\text{G}}(1) \end{aligned}$$

In terms of hills and valleys, both M-SERPT and Gittins have a hill age at 1, but M-SERPT has an additional hill age at $1-\delta$. But M-SERPT's extra hill age increases mean response time: a job of age $1-\delta$ will almost always finish with just δ more work, so it would be better to give those jobs priority over jobs at age 0. Gittins does not make this mistake.

[12] We note that Gittins is the solution for the special case where all jobs are present at the start, because without arrivals, any SOAP policy, including Gittins [6], acts like an MLPS policy based on its hills and valleys.

[13] RMLF resembles an MLPS policy, but it is not one because it uses randomization.

We now compute the mean response times of M-SERPT and Gittins for a system with job size distribution X. Suppose the load is $\rho = 1 - \varepsilon$, where $\varepsilon \in (0, 1)$ is small. We have

$$\begin{aligned} \bar{\rho}(0) &= 1 \quad \bar{\rho}(1-\delta) \approx \delta + \varepsilon \quad \bar{\rho}(1) \approx \delta + \varepsilon \quad \bar{\rho}(\infty) = \varepsilon \\ \tau(0) &= 0 \quad \tau(1-\delta) \approx \tfrac{1}{2} \qquad \tau(1) \approx \tfrac{1}{2} \qquad \tau(\infty) \approx 1, \end{aligned}$$

where the approximations assume $\delta, \varepsilon \ll 1$. By Propositions 4.7 and 4.8, the mean response time of M-SERPT is

$$\begin{aligned} \mathbf{E}[T_{\mathrm{MS}}] &\approx \mathbf{E}[T_{\mathrm{MS}}(1-\delta)] + \delta\mathbf{E}[T_{\mathrm{MS}}(1)] + \delta^2\mathbf{E}[T_{\mathrm{MS}}(\delta^{-1}+1)] \\ &= \left(\frac{\tau(1-\delta)}{\bar{\rho}(0)\cdot\bar{\rho}(1-\delta)} + \frac{1-\delta}{\bar{\rho}(0)}\right) + \delta\left(\frac{\tau(1)}{\bar{\rho}(1-\delta)\cdot\bar{\rho}(1)} + \frac{1}{\bar{\rho}(1-\delta)}\right) + \delta^2\left(\frac{\tau(\infty)}{\bar{\rho}(1)\cdot\bar{\rho}(\infty)} + \frac{\delta^{-1}+1}{\bar{\rho}(1)}\right) \\ &\approx \left(\frac{\frac{1}{2}}{\delta+\varepsilon} + 1 - \delta\right) + \delta\left(\frac{\frac{1}{2}}{(\delta+\varepsilon)^2} + \frac{1}{\delta+\varepsilon}\right) + \delta^2\left(\frac{1}{\varepsilon\cdot(\delta+\varepsilon)} + \frac{\delta^{-1}+1}{\delta+\varepsilon}\right) \\ &\approx \frac{1}{2(\delta+\varepsilon)}\left(\frac{2\delta+\varepsilon}{\delta+\varepsilon} + \frac{2\delta^2}{\varepsilon}\right). \end{aligned}$$

We now analyze the mean response time of Gittins. One can show using the full SOAP analysis [28] that when $\delta, \varepsilon \ll 1$, Propositions 4.7 and 4.8 give approximate equalities for Gittins, so

$$\begin{aligned} \mathbf{E}[T_{\mathrm{G}}] &\approx \mathbf{E}[T_{\mathrm{G}}(1-\delta)] + \delta\mathbf{E}[T_{\mathrm{G}}(1)] + \delta^2\mathbf{E}[T_{\mathrm{G}}(\delta^{-1}+1)] \\ &\approx \left(\frac{\tau(1)}{\bar{\rho}(0)\cdot\bar{\rho}(1)} + 1 - \delta\right) + \delta\left(\frac{\tau(1)}{\bar{\rho}(0)\cdot\bar{\rho}(1)} + 1\right) + \delta^2\left(\frac{\tau(\infty)}{\bar{\rho}(1)\cdot\bar{\rho}(\infty)} + \delta^{-1} + 1\right) \\ &\approx \left(\frac{\frac{1}{2}}{\delta+\varepsilon} + 1 - \delta\right) + \delta\left(\frac{\frac{1}{2}}{\delta+\varepsilon} + 1\right) + \delta^2\left(\frac{1}{\varepsilon\cdot(\delta+\varepsilon)} + \delta^{-1} + 1\right) \\ &\approx \frac{1}{2(\delta+\varepsilon)}\left(1 + \frac{2\delta^2}{\varepsilon}\right). \end{aligned}$$

This makes the mean response time ratio approximately

$$\frac{\mathbf{E}[T_{\mathrm{MS}}]}{\mathbf{E}[T_{\mathrm{G}}]} \approx \frac{2\delta^3 + 2\delta\varepsilon + \varepsilon^2}{2\delta^3 + \delta\varepsilon + \varepsilon^2}.$$

This ratio is at most 2, and it can approach 2 in any limit where the $\delta\varepsilon$ term dominates. This happens if we set $\varepsilon = \delta^{3/2}$ in the $\delta \to 0$ limit, so M-SERPT's approximation ratio is at least 2.

8 WHY CLOSING THE GAP IS HARD

In preliminary numerical studies, omitted for lack of space, we have computed the mean response time ratio between M-SERPT and Gittins for a variety of job size distributions. We have yet to observe a ratio greater than 2, with Section 7 describing the worst case we have found, motivating the following conjecture.

Conjecture 8.1. *For the problem of preemptive scheduling to minimize mean response time in an M/G/1 queue with unknown job sizes, the approximation ratio of M-SERPT is 2.*

The lower bound of 2 on M-SERPT's approximation ratio is less than the upper bound of 5 from Theorem 5.1. What would it take to close the gap? Recall from Fig. 5.1 that we prove Theorem 5.1 by combining the four following bounds. The main obstacle to closing the gap is that *each of the four bounds is tight in some setting.*

(i) Lemma 5.6 gives an upper bound on $\mathbf{E}[Q_{\mathrm{MS}}]/\mathbf{E}[Q_{\mathrm{G}}]$.
   – It is tight for the scenario described in Section 7.

(ii) Lemma 5.7 gives an upper bound on $\mathbf{E}[R_{\text{MS}}]$.
  – It is tight in the $\rho \to 1$ limit for Pareto job size distributions with shape parameter $\alpha \approx 1$ [17].
(iii) Proposition 4.9 gives a lower bound on $\mathbf{E}[R_{\text{G}}]$.
  – It is tight when Gittins is equivalent to FCFS, which occurs for some job size distributions [5].
(iv) Proposition 4.10 gives a lower bound on $\mathbf{E}[T_{\text{G}}]$.
  – It is tight in the $\rho \to 1$ limit for Pareto job size distributions with shape parameter $\alpha \approx 1$ [17].

The fact that each bound is tight means that tightening Theorem 5.1 requires new insight.

Although bounds (i)–(iv) are all tight, they are tight in *different settings*, meaning for different loads ρ and job size distributions X. This hints at a possible approach to tightening Theorem 5.1: we could refine bounds (i)–(iv) in a way that makes them more sensitive to the setting, especially the job size distribution. As an example of what this might mean, the settings in which bounds (i) and (iii) are tight have $\mathbf{Var}(X^2) < \infty$, while those in which bounds (ii) and (iv) are tight have $\mathbf{Var}(X^2) = \infty$. Thus, we might be able to improve on Theorem 5.1 if we refine each of bounds (i)–(iv) by "conditioning", meaning splitting into cases, on whether $\mathbf{Var}(X)$ is finite.

With that said, we suspect that refining bounds (i)–(iv) is more involved than simply conditioning on whether $\mathbf{Var}(x) = \infty$. In the rest of this section we review each bound, explain the settings in which they are tight in more detail, and discuss opportunities for refining or replacing them.

8.1 Tightening the M-SERPT Upper Bounds

We begin with bound (i), Lemma 5.6, which implies $\mathbf{E}[Q_{\text{MS}}] \le 2\mathbf{E}[Q_{\text{G}}]$. This bound is tight for the scenario described in Section 7. To find opportunities for tightening, recall that the proof of Lemma 5.6 works by looking at one valley at a time, showing a ratio bound for each valley separately. When proving the bound for valley $(u, v]$, we use the fact that $\bar{\rho}(u) \le 1$,[14] but this is tight for at most one valley. In the job size distribution from Section 7, nearly every job's size is in a valley with $\bar{\rho}(u) = 1$, which is why Lemma 5.6 is tight in that scenario. But many job size distributions do not have nearly all job sizes in one valley. We could perhaps refine Lemma 5.6 by conditioning on a parameter related to valleys, such as a bound $\zeta \in [0, 1]$ such that $\mathbf{P}\{X \in (u, v]\} \le \zeta$ for all valleys $(u, v]$.

Bound (ii), Lemma 5.7, says $\mathbf{E}[R_{\text{MS}}] \le \mathbf{E}[Q_{\text{MS}}] + \ell_\rho$, where

$$\ell_\rho = \left(\frac{1}{\rho} \log \frac{1}{1-\rho}\right) \mathbf{E}[X].$$

Lemma 5.7 can be tight in the $\rho \to 1$ limit when X has a Pareto job size distribution. For shape parameter $\alpha \in (1, 2)$, if $\bar{F}(x) = (1 + x)^{-\alpha}$, a result of Kamphorst and Zwart [17, Section 4.2.1] implies[15]

$$\begin{aligned} \mathbf{E}[Q_{\text{MS}}] &\approx \frac{\alpha(\alpha - 1)}{2 - \alpha} \cdot \ell_\rho \\ \mathbf{E}[R_{\text{MS}}] &\approx \alpha \cdot \ell_\rho \end{aligned} \tag{8.1}$$

as $\rho \to 1$. This means the tightness of Lemma 5.7 in the $\rho \to 1$ limit depends on α: it is tight for $\alpha \approx 1$ but extremely loose for $\alpha \approx 2$. Similar reasoning shows the bound is also loose for $\alpha > 2$ [17, Section 4.1.1]. This suggests that we could try to refine Lemma 5.7 by conditioning on the tail behavior of X. A concrete opportunity for tightening is in Step 1 of the proof: the difference between the two sides of the final inequality is $2\lambda \bar{F}(x)^2 \cdot \tau(x)/\bar{\rho}(x)^3$, whose contribution is negligible for $\alpha \approx 1$ but dominates for larger α [17]. Step 2 of the proof has a similar opportunity, but the

[14] Specifically, we apply Lemma 5.4 with $a = u$, and Lemma 5.4's proof uses $\bar{\rho}(a) \le 1$.
[15] Kamphorst and Zwart [17] consider the FB policy, but M-SERPT and FB are equivalent for this job size distribution because it has the IMRL property (Definition 6.1).

difference term is more complicated. Another obstacle to this approach is the lack of results in the style of Kamphorst and Zwart [17] that hold outside the $\rho \to 1$ limit.

8.2 Tightening the Gittins Lower Bounds

Bound (iii), Proposition 4.9, gives a trivial lower bound on Gittins's mean residence time, namely $\mathbf{E}[R_G] \geq \mathbf{E}[X]$. But even this trivial bound is tight for some job size distributions, namely those with the *new better than used in expectation* property [5]. This is because the Gittins policy is equivalent to FCFS for such distributions [5], and FCFS has mean residence time $\mathbf{E}[X]$. However, a result of Aalto et al. [6, Proposition 9] implies that if Gittins is equivalent to FCFS for some distribution X, then M-SERPT is also equivalent to FCFS. That is, when Gittins has very low residence time, so does M-SERPT. This hints that what we would really like is a direct bound on $\mathbf{E}[R_{MS}]/\mathbf{E}[R_G]$. Unfortunately, the residence time formula in Proposition 4.8 gives an upper bound on $\mathbf{E}[R_G]$, whereas we need a lower bound. Even if we could bound the gap between $\mathbf{E}[R_G]$ and the upper bound in Proposition 4.8, bounding $\mathbf{E}[R_{MS}]/\mathbf{E}[R_G]$ would likely still be at least as challenging as proving Lemma 5.6.

We finally turn to bound (iv), Proposition 4.10, which is a corollary of a result of Wierman et al. [30, Theorem 5.8]. It says $\mathbf{E}[T] \geq \ell_\rho$ for *any* scheduling policy, including size-based policies like SRPT. Despite this, by (8.1), Proposition 4.10 is tight in the $\rho \to 1$ limit when X has a Pareto job size distribution with shape parameter $\alpha \approx 1$. We are not aware of any other simple lower bound on SRPT's mean response time that holds for all job size distributions. One possibility for refining the bound would be to parametrize them along similar lines as further results of Wierman et al. [30, Theorems 5.4, 5.7, and 5.9]. Of course, we would prefer a bound that holds only for policies that, like Gittins, do not use job size information, but we suspect such a result requires new techniques.

9 CONCLUSION

We introduce M-SERPT, the first non-Gittins policy proven to achieve mean response time within a constant factor of Gittins's. Specifically, we show that M-SERPT is a 5-approximation of Gittins, with an even smaller approximation ratio at lower loads (Theorem 5.1). In addition to being an important result in its own right, the fact that M-SERPT has near-optimal mean response time resolves two open questions in M/G/1 scheduling theory (Section 6).

An open question is whether M-SERPT's approximation ratio is less than 5. We conjecture that the true approximation ratio is 2 (Conjecture 8.1). Another open question is how SERPT's mean response time compares to M-SERPT's. In preliminary numerical studies, we have observed very similar performance from SERPT and M-SERPT, with each sometimes outperforming the other, so we conjecture that SERPT is also a constant-factor approximation of Gittins.

ACKNOWLEDGMENTS

This work was supported by NSF-CSR-1763701, NSF-XPS-1629444 and a Microsoft Faculty Award 2018. Ziv Scully was supported by the NSF GRFP under grants DGE-1745016 and DGE-125222 and an ARCS Foundation scholarship. We thank the anonymous referees for their helpful comments.

REFERENCES

[1] Samuli Aalto and Urtzi Ayesta. 2006. Mean delay analysis of multi level processor sharing disciplines. In *INFOCOM 2006. 25th IEEE International Conference on Computer Communications. Proceedings*. IEEE, 1–11.

[2] S Aalto and U Ayesta. 2006. On the nonoptimality of the foreground-background discipline for IMRL service times. *Journal of Applied Probability* 43, 2 (2006), 523–534.

[3] Samuli Aalto, Urtzi Ayesta, Sem Borst, Vishal Misra, and Rudesindo Núñez-Queija. 2007. Beyond processor sharing. In *ACM SIGMETRICS Performance Evaluation Review*, Vol. 34. ACM, 36–43.

[4] Samuli Aalto, Urtzi Ayesta, and Eeva Nyberg-Oksanen. 2004. Two-level processor-sharing scheduling disciplines: mean delay analysis. In *ACM SIGMETRICS Performance Evaluation Review*, Vol. 32. ACM, 97–105.

[5] Samuli Aalto, Urtzi Ayesta, and Rhonda Righter. 2009. On the Gittins index in the M/G/1 queue. *Queueing Systems* 63, 1 (2009), 437–458.

[6] Samuli Aalto, Urtzi Ayesta, and Rhonda Righter. 2011. Properties of the Gittins index with application to optimal scheduling. *Probability in the Engineering and Informational Sciences* 25, 03 (2011), 269–288.

[7] Nikhil Bansal, Bart Kamphorst, and Bert Zwart. 2018. Achievable performance of blind policies in heavy traffic. *Mathematics of Operations Research* 43, 3 (2018), 949–964.

[8] Luca Becchetti and Stefano Leonardi. 2004. Nonclairvoyant scheduling to minimize the total flow time on single and parallel machines. *Journal of the ACM (JACM)* 51, 4 (2004), 517–539.

[9] Thomas Bonald and Alexandre Proutiere. 2002. Insensitivity in processor-sharing networks. *Performance Evaluation* 49, 1-4 (2002), 193–209.

[10] Jhelum Chakravorty and Aditya Mahajan. 2014. Multi-armed bandits, Gittins index, and its calculation. *Methods and applications of statistics in clinical trials: Planning, analysis, and inferential methods* 2 (2014), 416–435.

[11] Sing-Kong Cheung, Hans van den Berg, and Richard J Boucherie. 2006. Insensitive bounds for the moments of the sojourn time distribution in the M/G/1 processor-sharing queue. *Queueing systems* 53, 1-2 (2006), 7–18.

[12] Hanhua Feng and Vishal Misra. 2003. Mixed scheduling disciplines for network flows. In *ACM SIGMETRICS Performance Evaluation Review*, Vol. 31. ACM, 36–39.

[13] John C. Gittins, Kevin D. Glazebrook, and Richard Weber. 2011. *Multi-armed Bandit Allocation Indices*. John Wiley & Sons.

[14] Liang Guo and Ibrahim Matta. 2002. Scheduling flows with unknown sizes: Approximate analysis. In *ACM SIGMETRICS Performance Evaluation Review*, Vol. 30. ACM, 276–277.

[15] Mor Harchol-Balter. 2013. *Performance Modeling and Design of Computer Systems: Queueing Theory in Action* (1st ed.). Cambridge University Press, New York, NY, USA.

[16] Bala Kalyanasundaram and Kirk R Pruhs. 1997. Minimizing flow time nonclairvoyantly. In *Proceedings 38th Annual Symposium on Foundations of Computer Science*. IEEE, 345–352.

[17] Bart Kamphorst and Bert Zwart. 2017. Heavy-traffic analysis of sojourn time under the foreground-background scheduling policy. *arXiv preprint arXiv:1712.03853* (2017).

[18] Leonard Kleinrock. 1967. Time-shared systems: A theoretical treatment. *Journal of the ACM (JACM)* 14, 2 (1967), 242–261.

[19] Leonard Kleinrock. 1976. *Queueing Systems, Volume 2: Computer Applications*. Vol. 66. Wiley New York.

[20] Leonard Kleinrock and Richard R Muntz. 1972. Processor sharing queueing models of mixed scheduling disciplines for time shared system. *Journal of the ACM (JACM)* 19, 3 (1972), 464–482.

[21] Nicole Megow and Tjark Vredeveld. 2014. A Tight 2-Approximation for Preemptive Stochastic Scheduling. *Mathematics of Operations Research* 39, 4 (2014), 1297–1310.

[22] Rajeev Motwani, Steven Phillips, and Eric Torng. 1994. Nonclairvoyant scheduling. *Theoretical Computer Science* 130, 1 (1994), 17–47.

[23] Jayakrishnan Nair, Adam Wierman, and Bert Zwart. 2010. Tail-robust scheduling via limited processor sharing. *Performance Evaluation* 67, 11 (2010), 978–995.

[24] David Raz, Hanoch Levy, and Benjamin Avi-Itzhak. 2004. A resource-allocation queueing fairness measure. *ACM SIGMETRICS Performance Evaluation Review* 32, 1 (2004), 130–141.

[25] Rhonda Righter and J George Shanthikumar. 1989. Scheduling multiclass single server queueing systems to stochastically maximize the number of successful departures. *Probability in the Engineering and Informational Sciences* 3, 3 (1989), 323–333.

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

[27] Linus Schrage. 1968. A proof of the optimality of the shortest remaining processing time discipline. *Operations Research* 16, 3 (1968), 687–690.

[28] Ziv Scully, Mor Harchol-Balter, and Alan Scheller-Wolf. 2018. SOAP: One Clean Analysis of All Age-Based Scheduling Policies. *Proc. ACM Meas. Anal. Comput. Syst.* 2, 1, Article 16 (April 2018), 30 pages. https://doi.org/10.1145/3179419

[29] Adam Wierman. 2007. Fairness and classifications. In *ACM SIGMETRICS Performance Evaluation Review*, Vol. 34. ACM, 4–12.

[30] Adam Wierman, Mor Harchol-Balter, and Takayuki Osogami. 2005. Nearly insensitive bounds on SMART scheduling. In *ACM SIGMETRICS Performance Evaluation Review*, Vol. 33. ACM, 205–216.

[31] Genji Yamazaki and Hirotaka Sakasegawa. 1987. An optimal design problem for limited processor sharing systems. *Management Science* 33, 8 (1987), 1010–1019.

A NO APPROXIMATION RATIO FOR TRADITIONAL POLICIES

In this appendix we discuss the performance of three traditional policies: FCFS, FB, and PS. We will show that none of these policies are constant-factor approximations for mean response time. That is, the ratio of each policy's mean response times to that of Gittins can be unboundedly large.

FCFS has mean response time [15]

$$\mathbf{E}[T_{\text{FCFS}}] = \frac{\lambda \mathbf{E}[X^2]}{2(1-\rho)} + \mathbf{E}[X].$$

This is infinite if X has infinite variance, but other policies have finite mean response time for all job size distributions, so FCFS has no constant-factor approximation ratio.

For the specific case where all jobs have size x, FB has mean response time [15]

$$\mathbf{E}[T_{\text{FB}}] = \frac{\lambda x^2}{2(1-\rho)^2} + \frac{x}{1-\rho}.$$

This is worse than FCFS's mean response time in the same case by a factor of $1/(1-\rho)$, which becomes arbitrarily large in the $\rho \to 1$ limit, so FB has no constant-factor approximation ratio.

PS has mean response time [15]

$$\mathbf{E}[T_{\text{PS}}] = \frac{\mathbf{E}[X]}{1-\rho}.$$

That is, the response time of PS is insensitive to the details of the job size distribution, depending only on the mean. While PS is thus generally considered to have reasonable performance for all job size distributions, there are certain distributions where other policies outperform PS by far. For example, Kamphorst and Zwart [17] show that when X is a Pareto distribution with shape parameter $\alpha \in (1, 2)$, FB has mean response time that scales as

$$\mathbf{E}[T_{\text{FB}}] \approx \frac{\alpha \mathbf{E}[X]}{2-\alpha} \log \frac{1}{1-\rho}$$

in the $\rho \to 1$ limit. Thus, the mean response time ratio between PS and FB becomes arbitrarily large in the $\rho \to 1$ limit, so PS has no constant-factor approximation ratio.

B DIFFICULTY OF COMPUTING THE GITTINS POLICY

In this appendix we discuss in more detail why it is difficult to compute the Gittins rank function. We begin with the simpler case of discrete job size distributions (Appendix B.1) before turning to continuous job size distributions (Appendix B.2).

B.1 Discrete Job Size Distributions

All the algorithms discussed in this section assume input in the form of a list of (x, p) pairs sorted by x, where x is a support point and p is the probability of outcome x.

The problem of computing the Gittins rank[16] of all states in a finite Markov chains is a well studied problem for which the best known algorithms take $O(n^3)$ time, where n is the number of states in the Markov chain [10]. The reader may recall that we claim in Table 1.1 that Gittins takes $O(n^2)$ time to compute. This is due to two discrepancies between algorithms in the literature and the problem we consider, namely computing the Gittins rank function for a discrete job size distribution.

- Algorithms in the literature assume an arbitrary finite Markov chain. However, a discrete job size distribution has a very simple structure when viewed as a Markov chain. Each support point is a state, and each has only two transitions with nonzero probability: to the next

[16] Most literature refers to the *Gittins index*, which is simply the reciprocal of the Gittins rank.

support point and to a terminal state.[17] In this respect, our problem is *easier* than the one solved in the literature.
- Algorithms in the literature compute the Gittins rank at each state, which in our case corresponds to each support point. However, the full Gittins rank function assigns ranks to all ages, and ages between adjacent support points are not covered by algorithms in the literature. In this respect, our problem is *harder* than the one solved in the literature.

It turns out that the former difference has the greater impact. Specifically, if one uses sparse matrix operations, algorithms in the literature can be implemented such that they take only $O(n^2)$ time [10], because the Markov chain of a discrete job size distribution has only $O(n)$ transitions with nonzero probability. The output of this algorithm is the Gittins rank of each support point, but it remains to compute the rank function at other ages. Between each pair of adjacent support points, the Gittins rank function is piecewise linear with at most $O(n)$ segments. This means a post-processing step taking $O(n)$ time per support point, and thus $O(n^2)$ time total, can fill in the gaps between adjacent support points.

We have summarized how to use state-of-the-art algorithms from the literature to compute the Gittins rank function in $O(n^2)$ time. Whether there exists an algorithm computing the Gittins rank function in $o(n^2)$ time remains an open problem.

Finally, we briefly sketch an algorithm that computes the SERPT and M-SERPT rank functions in $O(n)$ time. Computing $r_S(x)$ at each support point x can be done with a table containing $\overline{F}(x)$ and $\int_x^\infty \overline{F}(t)\,\mathrm{d}t$ for each support point x, which can be generated with scans that take $O(n)$ time each. This yields the SERPT rank at each support point, and an additional $O(n)$ scan yields the same for M-SERPT. Between adjacent support points, SERPT's rank function simply decreases at slope 1 while M-SERPT's is constant.

B.2 Continuous Job Size Distributions

The Gittins policy for continuous job size distributions has received some attention, with results characterizing the Gittins rank function available under various assumptions on the job size distribution [5, 6]. However, none of the prior work explicitly addresses computing the Gittins policy for a general continuous job size distribution. Here we review the most general characterization result and show why it does not solve the problem of computing the Gittins rank function.

Aalto et al. [6, Propositions 1 and 11] show the following result. Suppose there exist ages $0 = v_0, u_1, v_1, u_2, v_2, \ldots$ such that for all $i \geq 1$, the job size distribution's hazard rate h is

- strictly decreasing for (u_i, v_i) and
- increasing for (v_{i-1}, u_i).

Then for all $i \geq 1$, there exists an age $w_i \in [u_i, v_i]$ such that the Gittins rank function r_G is

- strictly increasing for (u_i, v_i) and
- decreasing for (w_{i-1}, u_i).[18]

Knowing something about the monotonicity of the Gittins rank function is potentially helpful for computing it. However, the results of Aalto et al. [6] do not provide a way to *compute* the critical ages w_i. Moreover, even if we could compute the ages w_i, as we explain below, computing the rank function can be at least as hard as in the discrete case.

For each age a, there is an optimal stopping age $b_*(a)$ that solves the optimization problem in $r_G(a)$ (Definition 2.1). We know by results of Aalto et al. [6] that if $b_*(a) > a$, then $b_*(a)$ lies in

[17] The terminal state is the maximum support point. Additionally, there is an initial state at age 0. In the following discussion, any mention of "adjacent support points" also applies to the interval between 0 and the first support point.
[18] We define $w_0 = 0$.

interval $[u_i, w_i]$ for some i, but we do not know which i. This makes the search for $b_*(a)$ intractable if there are infinitely many intervals $[u_i, w_i]$ and at least as hard as the discrete case if there are finitely many.

C SOAP MEAN RESPONSE TIME USING HILLS AND VALLEYS

Propositions 4.7 and 4.8 follow immediately from results of Scully et al. [28, Theorem 5.5, see also Lemmas 5.2 and 5.3]. The main obstacle is a difference in notation. Below we translate from the notation in our paper to the notation of Scully et al. [28]:

$$\bar{\rho}(y(x)) = 1 - \rho^{\text{new}}[r_x^{\text{worst}}(0)] \geq 1 - \rho^{\text{new}}[r_x^{\text{worst}}(a)]$$
$$\bar{\rho}(z(x)) = 1 - \rho_0^{\text{old}}[r_x^{\text{worst}}(0)]$$
$$\tau(z(x)) = \frac{\lambda}{2}\mathbf{E}[X_0^{\text{old}}[r_x^{\text{worst}}(0)]] \leq \frac{\lambda}{2}\sum_{i=0}^{\infty}\mathbf{E}[X_i^{\text{old}}[r_x^{\text{worst}}(0)]].$$

When the rank function is monotonic, showing that the bounds in Propositions 4.7 and 4.8 become equalities boils down to proving that the two inequalities above become equalities. We first note that any decreasing rank function is equivalent to FCFS. But FCFS can also be expressed by a constant rank function, which is weakly increasing. We therefore restrict our attention to increasing rank functions, for which the following properties are easily shown:

- $r_x^{\text{worst}}(a) = r_x^{\text{worst}}(0)$ for all ages a [28, Definition 4.1], and
- $X_i^{\text{old}}[r] = 0$ with probability 1 for all ranks r and integers $i \geq 1$. [28, Definition 4.3].

Thus, both inequalities above become equalities for monotonic rank functions.

D DEFERRED PROOFS

Theorem 5.1. *The mean response time ratio between M-SERPT and Gittins is bounded by*

$$\frac{\mathbf{E}[T_{MS}]}{\mathbf{E}[T_G]} \leq \begin{cases} \dfrac{4}{1+\sqrt{1-\rho}} & 0 \leq \rho < 0.9587 \\ \dfrac{1}{\rho}\log\dfrac{1}{1-\rho} & 0.9587 \leq \rho < 0.9898 \\ 1 + \dfrac{4}{1+\sqrt{1-\rho}} & 0.9898 \leq \rho < 1. \end{cases}$$

Proof. Bounding mean response time amounts to bounding mean waiting and residence times. By Lemma 5.6,

$$\mathbf{E}[Q_{\text{MS}}] \leq \frac{2}{1+\sqrt{1-\rho}}\mathbf{E}[Q_{\text{G}}],$$

and by Lemma 5.7,

$$\mathbf{E}[R_{\text{MS}}] \leq \mathbf{E}[Q_{\text{MS}}] + \left(\frac{1}{\rho}\log\frac{1}{1-\rho}\right)\mathbf{E}[X]. \tag{D.1}$$

We can give two different bounds on the last term of (D.1), each of which yields a bound on the mean response time ratio. Applying Proposition 4.9 yields

$$\frac{\mathbf{E}[T_{\text{MS}}]}{\mathbf{E}[T_{\text{G}}]} \leq \frac{\frac{4}{1+\sqrt{1-\rho}}\mathbf{E}[Q_{\text{G}}] + \left(\frac{1}{\rho}\log\frac{1}{1-\rho}\right)\mathbf{E}[R_{\text{G}}]}{\mathbf{E}[Q_{\text{G}}] + \mathbf{E}[R_{\text{G}}]}$$
$$\leq \max\left\{\frac{4}{1+\sqrt{1-\rho}}, \frac{1}{\rho}\log\frac{1}{1-\rho}\right\}.$$

Applying Proposition 4.10 instead yields

$$\frac{\mathbf{E}[T_{\text{MS}}]}{\mathbf{E}[T_{\text{G}}]} \le \frac{\left(1 + \frac{4}{1+\sqrt{1-\rho}}\right)\mathbf{E}[Q_{\text{G}}] + \mathbf{E}[R_{\text{G}}]}{\mathbf{E}[Q_{\text{G}}] + \mathbf{E}[R_{\text{G}}]} \le 1 + \frac{4}{1+\sqrt{1-\rho}}.$$

Taking the minimum of these two bounds gives us

$$\frac{\mathbf{E}[T_{\text{MS}}]}{\mathbf{E}[T_{\text{G}}]} \le \min\left\{\max\left\{\frac{4}{1+\sqrt{1-\rho}}, \frac{1}{\rho}\log\frac{1}{1-\rho}\right\}, 1 + \frac{4}{1+\sqrt{1-\rho}}\right\},$$

which expands to the desired piecewise bound. □

Lemma 5.3. *Every Gittins hill age is also an M-SERPT hill age, and similarly for hill sizes.*

Proof. We prove the result for hill ages. The corresponding result for hill sizes then follows immediately from the observation that x is a hill size if and only if there exists $\varepsilon > 0$ such that all ages in $[x, x+\varepsilon)$ are hill ages, so we can simply apply the hill age result to those intervals.

It is immediate from Definition 2.4 that SERPT and M-SERPT have the same hill ages, so in this proof, we work with SERPT instead of M-SERPT.

At the core of our argument is the following definition. For ages $a < b$, let

$$\begin{aligned}
\eta(a,b) &= \frac{\int_a^b \overline{F}(t)\,\mathrm{d}t}{\overline{F}(a) - \overline{F}(b)} \\
\eta(a,a) &= \lim_{b\to a}\eta(a,b) = \frac{\overline{F}(a)}{f(a)} = \frac{1}{h(a)} \\
\eta(a,\infty) &= \lim_{b\to\infty}\eta(a,b) = \mathbf{E}[X - a \mid X > a].
\end{aligned}$$

The function η is a version of the *efficiency function* commonly used in the M/G/1 Gittins policy literature [5, 6]. Its continuity is inherited from the fact that X has a density function (Section 2). It is closely related to the rank functions of SERPT and Gittins:[19]

$$\begin{aligned}
r_{\text{S}}(a) &= \eta(a,\infty) \\
r_{\text{G}}(a) &= \min_{b\ge a}\eta(a,b) \le r_{\text{S}}(a)
\end{aligned} \tag{D.2}$$

It is simple to verify that for any ages $a \le b \le c$,

$$\begin{array}{ccc}
\eta(a,b) \le \eta(a,c) & \le & \eta(b,c) \\
\Updownarrow & & \\
\eta(a,b) \le \eta(a,c) & & \\
\Updownarrow & & \\
\eta(a,c) & \le & \eta(b,c) \\
\Updownarrow & & \\
\eta(a,b) & \le & \eta(b,c).
\end{array} \tag{D.3}$$

and similarly for strict inequalities when $a < b < c$.

A useful intuition is that $\eta(a,b)$ gives a "score" to the interval $[a,b]$, where lower scores are better. SERPT gives a job at age a rank equal to the score of $[a,\infty]$, while Gittins is pickier, choosing the best score among all intervals that start at a. What (D.3) says is that if we divide an interval into two pieces, the score of the interval is between scores of its pieces.

[19] The minimum in $r_{\text{G}}(a)$ always exists because we allow $b = a$ and $b = \infty$.

Let v be a Gittins hill age and consider any age $u < v$. We want to show that v is a SERPT hill age, which amounts to showing $r_S(u) < r_S(v)$. By (D.2) and (D.3), it suffices to show

$$\eta(u, v) \le r_G(v). \tag{D.4}$$

For simplicity, we show (D.4) only for $u = 0$, explaining at the end of the proof why we can do so without loss of generality.

To show (D.4) with $u = 0$, we need to understand $\eta(0, v)$. We can partition $[0, v]$ into Gittins hills and valleys, meaning there exist

$$0 = z_0 \le y_1 < z_1 < \ldots < y_n < z_n \le y_{n+1} = v$$

such that

- $(y_i, z_i]$ is a Gittins valley for all $i \in \{1, \ldots, n\}$,
- $(z_i, y_{i+1}]$ is a Gittins hill for all $i \in \{1, \ldots, n\}$, and
- either $z_0 = y_1$ or $(z_0, y_1]$ is a Gittins hill.

By repeatedly applying (D.3), it suffices to show that for each hill $(z_i, y_{i+1}]$,

$$\eta(z_i, y_{i+1}) < r_G(v), \tag{D.5}$$

and that for each valley $(y_i, z_i]$,

$$\eta(y_i, z_i) < r_G(v). \tag{D.6}$$

We prove these bounds in Steps 1 and 2 below, respectively.

Step 1: bound for Gittins hills. Let $(z_i, y_{i+1}]$ be a Gittins hill. Continuity of r_G (Section 2.1) and a result of Aalto et al. [6, Proposition 3] together imply that for all $a \in [z_i, y_{i+1})$,

$$r_G(a) = \frac{1}{h(a)} < \frac{1}{h(y_{i+1})} = r_G(y_{i+1}),$$

from which another result [6, Lemma 5] yields

$$r_G(z_i) = \eta(z_i, z_i) \le \eta(z_i, y_{i+1}).$$

By (D.3), we also have

$$\eta(z_i, y_{i+1}) \le \eta(y_{i+1}, y_{i+1}) = r_G(y_{i+1})$$

Combining this with the fact that $v > y_{i+1}$ is a Gittins hill age implies (D.5), as desired.

Step 2: bound for Gittins valleys. Let $(y_i, z_i]$ be a Gittins valley. A fundamental property of Gittins [13, Lemma 2.2] implies[20]

$$r_G(y_i) = \eta(y_i, z_i).$$

Combining this with the fact that $v > y_i$ is a Gittins hill age implies (D.6), as desired.

With Steps 1 and 2 we have shown (D.4) for $u = 0$. To generalize the argument to $u > 0$, we observe that the rank functions of SERPT and Gittins at ages u and later do not depend on ages earlier than u. Consider a modified job size distribution $X' = (X - u \mid X > u)$. Writing r' for rank functions with distribution X', we have

$$r'_S(a) = r_S(a + u)$$
$$r'_G(a) = r_G(a + u)$$

for all ages a. Switching job size distributions from X to X' simply shifts the rank functions by u, so $v - u$ is a Gittins hill age for X'. This transforms the $u > 0$ case for X into the $u = 0$ case for X'. □

Received October 2019; revised December 2019; accepted January 2020

[20] Gittins et al. [13] focus on a discrete setting, but essentially the same proof holds in our continuous setting.