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

Ziv Scully
Carnegie Mellon University
Computer Science Department
Pittsburgh, PA, USA
zscully@cs.cmu.edu

Mor Harchol-Balter
Carnegie Mellon University
Computer Science Department
Pittsburgh, PA, USA
harchol@cs.cmu.edu

Alan Scheller-Wolf
Carnegie Mellon University
Tepper School of Business
Pittsburgh, PA, USA
awolf@andrew.cmu.edu

ABSTRACT

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*.

KEYWORDS

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. In *ACM SIGMETRICS / International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS '20 Abstracts), June 8–12, 2020, Boston, MA, USA.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3393691.3394216

SIGMETRICS '20 Abstracts, June 8–12, 2020, Boston, MA, USA

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. 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.

When individual job sizes are unknown but the job size distribution is known, the *Gittins* policy minimizes mean response time. 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 [1].
- 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, 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, such as *first-come, first-serve* (FCFS), *foreground-background* (FB), and *processor sharing* (PS). While each of these policies performs well for *some* job size distributions, there are no guarantees of near-optimal mean response time for any non-Gittins policy that hold across *all* job size distributions. 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 to compute than Gittins for both discrete and continuous job size distributions [2, Appendix B], as summarized in Table 1.1.

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

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 [1], which can analyze the mean response time of any scheduling policy specified by a rank function. We can use SOAP to *numerically* compute $C_{\text{SERPT}}(X)$ for any given job size distribution X, but SOAP does not bound 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 [2, 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.

We introduce *monotonic SERPT* (M-SERPT), a new policy that is simple to compute and has provably near-optimal mean response time. M-SERPT's rank function is like SERPT's, except *a job's rank never improves:*

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

The monotonicity of M-SERPT's rank function is what makes its mean response time expression tractable.

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 [2, Theorem 5.1]. 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 lower 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. M-SERPT's performance is often equal or very close to Gittins's. For example, Figure 1.1 compares the mean response times of several policies, including M-SERPT, to that of Gittins,

[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.

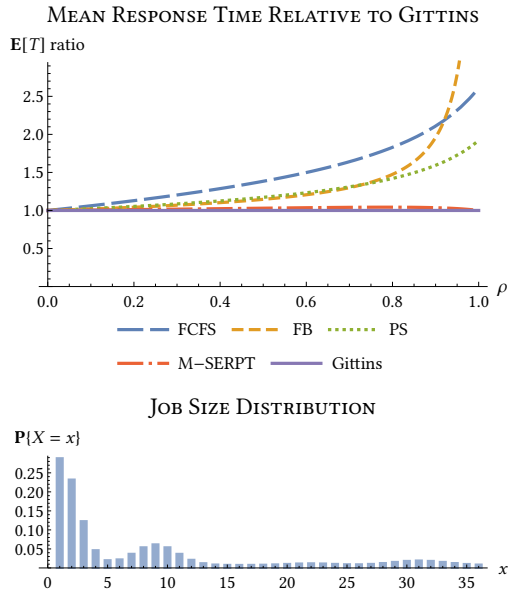

Figure 1.1: Mean Response Time Comparison

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. In further preliminary numerical experiments, we only observed a mean response time difference of more than 15% in a specific pathological scenario [2, Section 6].

Our paper [2] makes the following specific contributions:

- We define the *monotonic SERPT* (M-SERPT) policy, a new variant of SERPT.
- We introduce a new simplification of the SOAP response time analysis that yields a tractable mean response time expression for M-SERPT.
- 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.
- We use the fact that M-SERPT is a 5-approximation to resolve two open questions in M/G/1 scheduling theory. One concerns FB's performance for job size distributions with the *increasing mean residual lifetime* (IMRL) property, and the other characterizes the performance achievable by *multilevel processor sharing* (MLPS) policies.

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] 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

[2] 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