The Gittins Policy is Nearly Optimal in the M/G/k under Extremely General Conditions

ZIV SCULLY, Carnegie Mellon University, USA
ISAAC GROSOF, Carnegie Mellon University, USA
MOR HARCHOL-BALTER, Carnegie Mellon University, USA

The Gittins scheduling policy minimizes mean response time in the preemptive M/G/1 queue in a wide variety of settings. Most famously, Gittins is optimal when service requirements are unknown but drawn from a known distribution. Gittins is also optimal much more generally, adapting to any amount of available information. However, scheduling to minimize mean response time in a multiserver setting, specifically the central-queue M/G/k, is a much more difficult problem.

In this work we give the first general analysis of Gittins in the M/G/k. Specifically, we show that under extremely general conditions, Gittins's mean response time in the M/G/k is at most its mean response time in the M/G/1 plus an $O(k \log(1/(1-\rho)))$ additive term, where ρ is the system load. A consequence of this result is that Gittins is heavy-traffic optimal in the M/G/k if the service requirement distribution S satisfies $\mathbf{E}[S^2(\log S)^+] < \infty$. This is the most general result on minimizing mean response time in the M/G/k to date.

To prove our results, we combine properties of the Gittins policy and Palm calculus in a novel way. Notably, our technique overcomes the limitations of tagged job methods used in prior scheduling analyses.

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/k; response time; latency; sojourn time; Gittins policy; heavy traffic; Markov process

ACM Reference Format:
Ziv Scully, Isaac Grosof, and Mor Harchol-Balter. 2020. The Gittins Policy is Nearly Optimal in the M/G/k under Extremely General Conditions. *Proc. ACM Meas. Anal. Comput. Syst.* 4, 3, Article 43 (December 2020), 29 pages. https://doi.org/10.1145/3428328

1 INTRODUCTION

The question of how to schedule jobs so as to minimize mean response time is a classic question in the queueing literature. Here the response time of a job is the time from when the job arrives until it completes service, and the goal is to minimize the average response time across all jobs. In the M/G/1 queue, the answer is the *Gittins* policy. Gittins minimizes mean response time in a huge variety of settings [7]. These include the case where one has perfect information about job service

Authors' addresses: Ziv Scully, Carnegie Mellon University, Computer Science Department, 5000 Forbes Ave, Pittsburgh, PA, 15213, USA, zscully@cs.cmu.edu; Isaac Grosof, Carnegie Mellon University, Computer Science Department, 5000 Forbes Ave, Pittsburgh, PA, 15213, USA, igrosof@cs.cmu.edu; Mor Harchol-Balter, Carnegie Mellon University, Computer Science Department, 5000 Forbes Ave, Pittsburgh, PA, 15213, USA, harchol@cs.cmu.edu.

2476-1249/2020/12-ART43 $15.00
https://doi.org/10.1145/3428328

requirements (sizes); the case where one has zero information about job sizes, but one knows the distribution of the job sizes; and cases where one has partial knowledge about jobs' sizes.

Gittins is an index policy which assigns a "rank" to each job, where at every moment of time the job served is the one with lowest rank. Roughy speaking, a job has low rank if it is likely to complete soon (see Section 4). Note that in case where one has perfect information about job sizes, the Gittins policy is equivalent to the Shortest-Remaining-Processing-Time (SRPT) policy.

Unfortunately, minimizing mean response time in multiserver systems like the M/G/k is much harder. In the case where one has perfect information about job sizes, it was recently shown that SRPT, which is optimal for the M/G/1, is also optimal for the M/G/k in heavy traffic and near-optimal at all loads [10]. Here "near-optimal" specifically means that the response time differs from optimal by an $O(k \log(1/(1-\rho)))$ additive term, where $\rho < 1$ is the system load. One might therefore hope that in the case where job sizes are not known, or not fully known, that the Gittins policy, which is optimal for the M/G/1, is also optimal for the M/G/k.

In this paper, we show that the Gittins policy in the M/G/k is optimal in heavy traffic, meaning the $\rho \to 1$ limit, and near-optimal at all loads. Here "near-optimal" again means within an $O(k \log(1/(1-\rho)))$ additive term of optimal. We do so by comparing the Gittins policy in the M/G/k to the Gittins policy in the M/G/1, where it is known to be optimal. Our approach involves applying a novel Palm calculus technique to bounding mean response time in the M/G/k, directly relating response time to certain quantities of steady-state work. Our Palm calculus technique leverages key properties of the Gittins rank function which are responsible for Gittins' optimality in the M/G/1. In contrast, recent prior work on analyzing scheduling in the M/G/k has focused on tagged-job methods [10, 18], which are not well-suited to analyzing Gittins (see Section 6).

Our analysis allows us to prove Theorem 1.1, which upper-bounds the *gap* between the mean response time of the Gittins policy in the M/G/k and the M/G/1, where the server in the M/G/1 is k times faster than the servers in the M/G/k. This is important because the mean response time in the M/G/1 under Gittins is a lower bound on the mean response time in the M/G/k under any scheduling policy. In the common case where the variance of the service requirement distribution is finite, Theorem 1.2 gives a simpler bound on the mean response time gap. The tightness of our bound enables us to prove in Theorem 1.3 that Gittins is heavy-traffic optimal in the M/G/k.

Our results hold under an extremely broad job model (see Section 3) and under a very permissive condition on the service requirement distribution. Both the job model and the condition on the service requirement distribution are far broader than typically assumed [9, 10, 18].

Our extremely broad job model allows the information that is revealed as a job runs to be very complex and general. As a basic example, the job could reveal partial information about its service requirement upon arrival, such as a noisy estimate of the service requirement. Beyond that, each job could be broken into a series of stages, with information revealed whenever a stage completes. We even support scenarios where service requirement information is revealed continuously over time as a job runs. In general, we allow a job to consist of an arbitrary absorbing Markov process, revealing information according to the evolution of the state (see Section 3.2).

The condition we require on the service requirement distribution are also extremely general. For example, Theorem 1.1, which proves the near-optimality of Gittins in the M/G/k, requires only that the service requirement distribution have a finite αth moment for some $\alpha > 1$. Moreover, this condition is independent of the underlying job Markov process.

1.1 Main Results

Our main results concern the M/G/k queue with arrival rate λ and service requirement distribution S. All three theorems compare a k-server system with a single-server system with the total service

capacity 1, meaning each server has speed $1/k$. We denote the load by $\rho = \lambda \mathbf{E}[S]$ and the mean response time in the M/G/k by $\mathbf{E}[T_k]$. See Section 3 for a full description of the system model.

Theorem 1.1. *For all $\alpha > 1$, if $\mathbf{E}[S^\alpha] < \infty$, then the mean response time gap between the M/G/k and M/G/1 under Gittins is at most*

$$\mathbf{E}[T_k] - \mathbf{E}[T_1] \leq (k-1)\,\mathbf{E}[S]\left(\frac{1}{\alpha-1}\left(\log\frac{1}{1-\rho} + \log\frac{\mathbf{E}[S^\alpha]}{\alpha\,\mathbf{E}[S]^\alpha} + 1\right) + 4.547\right).$$

Theorem 1.2. *If $C_S^2 = \mathbf{Var}[S^2]/\mathbf{E}[S]^2 < \infty$, then the mean response time gap between the M/G/k and M/G/1 under Gittins is at most*

$$\mathbf{E}[T_k] - \mathbf{E}[T_1] \leq (k-1)\,\mathbf{E}[S]\left(\log\frac{1}{1-\rho} + \log(1+C_S^2) + 4.811\right).$$

Theorem 1.3. *If $\mathbf{E}[S^2(\log S)^+] < \infty$, then under Gittins,*

$$\lim_{\rho\to 1}\frac{\mathbf{E}[T_k]}{\mathbf{E}[T_1]} = 1,$$

so Gittins minimizes mean response time in the M/G/k in the heavy-traffic limit.

1.2 Contributions and Outline

Our main result is Theorem 1.1, where we prove that Gittins is near-optimal in the M/G/k. As a corollary of this result, in Theorem 1.3, we prove that Gittins is heavy-traffic optimal for the M/G/k. Along the way to proving these results, we make several contributions of independent interest. The first part of the paper introduces our model.

- In Section 3, we lay out an extremely general job model, allowing a highly flexible handling of the information that is revealed as a job runs.
- In Section 4, we give a treatment of the Gittins policy under our highly general job model.

The second part of the paper proves our main results.

- In Section 5, we introduce the "Gittins game", a framework for understanding the Gittins policy which we use throughout the paper to prove crucial properties of the policy.
- In Section 6, we derive a new formula for mean response time in the M/G/k in terms of the mean amount of different "relevant" subsets of work in the system.
- In Section 7, we present a new decomposition law that bounds the gap between the mean relevant work in the M/G/k and the M/G/1.
  - In addition to being useful in the M/G/k, our techniques also yield an elegant new proof of the optimality of Gittins in the M/G/1 (Theorem 7.3).
- Finally, in Section 8, we prove our main results.

Appendix A contains a summary of notation.

2 PRIOR WORK

There are several prior results on near-optimal scheduling in the preemptive M/G/k. The most general such results are the following:

- Grosof et al. [10] study the perfect-information case where service requirements are exactly known. They show that SRPT, which is equivalent to Gittins in this setting, is near-optimal and heavy-traffic optimal.
- Scully et al. [18] study the zero-information case where only the service requirement distribution is known. They show that a variation on Gittins called Monotonic Gittins (M-Gittins) is heavy-traffic optimal.

- Glazebrook and Niño-Mora [9] study the multiclass M/M/k with Bernoulli feedback. In this setting, a job's class gives partial information about its remaining service requirement. They show that Gittins is near-optimal and heavy-traffic optimal in this setting.

Our setting (Section 3) is strictly more general than each of the above settings. With that said, the specificity of the above settings allows results that are sometimes stronger than ours. We compare our results to each of the above in turn in Section 2.1, discussing differences in our techniques in Section 2.2. We also compare to a related result in nonpreemptive scheduling in Section 2.3.

2.1 Preemptive Scheduling

2.1.1 Perfect Information. Grosof et al. [10] show that the mean response time gap between the M/G/k and M/G/1 under SRPT is at most

$$\mathbf{E}[T_k] - \mathbf{E}[T_1] \le \frac{(2k-1)\,\mathbf{E}[S]}{\rho} \log \frac{1}{1-\rho}. \tag{2.1}$$

They then use this result to show that SRPT is heavy-traffic optimal under a condition which is slightly stronger than the existence of $\alpha > 2$ such that $\mathbf{E}[S^\alpha] < \infty$.

Our Theorem 1.1 is sometimes stronger and sometimes weaker than (2.1), depending on the service requirement distribution S, while our Theorem 1.3 is strictly stronger than the SRPT heavy-traffic result.[1]

2.1.2 Zero Information. Scully et al. [18] introduce a variant of Gittins called M-Gittins. They show that under a complex but relatively permissive condition on the service requirement distribution S, the mean response time gap between the M/G/k and M/G/1 under M-Gittins is, for all $\varepsilon > 0$,

$$\mathbf{E}[T_k] - \mathbf{E}[T_1] = O(k) \cdot o\left(\frac{1}{(1-\rho)^\varepsilon}\right). \tag{2.2}$$

They then use this result to show that under the same complex condition on S, if additionally $\mathbf{Var}[S] < \infty$, then M-Gittins is heavy-traffic optimal.

Our Theorem 1.1 differs from the results of Scully et al. [18] in several ways. The most obvious difference is that our bounds are explicit, whereas (2.2) gives only the asymptotic order of growth. Another difference is that our result is much more general. While Scully et al. [18] study only the zero-information case and require S to satisfy extra conditions, our model is much more general than the zero-information case, and Theorem 1.1 requires only $\mathbf{E}[S^\alpha] < \infty$ for some $\alpha > 1$. Finally, due to limitations in their technique (see Section 2.2), Scully et al. [18] analyze M-Gittins instead of Gittins. Our techniques overcome these limitations, allowing us to analyze Gittins.

2.1.3 Multiclass M/M/k with Bernoulli Feedback. Glazebrook and Niño-Mora [9] study a multiclass M/M/k with Bernoulli feedback and finitely many job classes. They show that the mean response time gap between the M/M/k and M/M/1 under Gittins is at most

$$\mathbf{E}[T_k] - \mathbf{E}[T_1] = O(k), \tag{2.3}$$

which immediately implies heavy-traffic optimality of Gittins. This is a stronger result than Theorem 1.1, but our results hold under much more general conditions, as explained below.

One interpretation of the multiclass M/M/k with Bernoulli feedback is that each job is a finite-state absorbing Markov chain: the states are job classes, plus one absorbing state indicating completion, and the transition rates are determined by the service rates and feedback probabilities. One can think of this setting as a simple multistage job model, as described in Section 3.1.

[1] With this said, our proof of Theorem 1.3 requires only that $\mathbf{E}[T_k] - \mathbf{E}[T_1] = O(k \log(1/(1-\rho)))$, so for the case of SRPT, one could use (2.1) as a starting point.

Our job model strictly generalizes the Markov chain model described above. Specifically, we model jobs as arbitrary absorbing Markov processes (Section 3.2), which is much more general than finite-state Markov chains. The techniques used by Glazebrook and Niño-Mora [9] are not easy to generalize to infinite-state systems (Section 2.2). Handling an infinite-state job model is one of the main obstacles we overcome in our proofs.

2.2 Differences in Proof Techniques

The techniques Grosof et al. [10] and Scully et al. [18] use in their respective proofs of (2.1) and (2.2) are very different from the techniques we use to prove Theorem 1.1. Specifically, they both use the "tagged job" method [11], bounding the amount of "relevant work" that is completed while a random job is in the system. While the tagged job method applies to a very wide variety of scheduling policies in the M/G/1 [20], it has only been successfully applied to the M/G/k for a much smaller set of scheduling policies. Scully et al. [18, Appendix A] describe obstacles to using the tagged job method in the M/G/k, explaining that the method as it currently exists is unlikely to yield good results for Gittins, even in just the zero-information case. We overcome these obstacles by avoiding the tagged job method entirely.

Glazebrook and Niño-Mora [9] analyze the multiclass M/M/k under Gittins using a technique that is similar to ours, but in a much less general setting. Specifically, in their model, jobs are *finite-state* Markov chains (Section 2.1.3), which simplifies their proof in two ways:

- Several steps of their proof use finite-dimensional matrices, where the dimensions correspond to the size of the job state space. Our proofs avoid reasoning about individual states, leading to more general results, sometimes with a more elegant proof.
- Because the job state space is finite, there exists a uniform bound $s_{\max}$ such that the expected remaining service requirement of a job is at most $s_{\max}$, regardless of the job's current state. This is the reason that the mean response time gap in (2.3) does not depend on load ρ. One of the main obstacles we overcome in our proof is bounding the mean response time gap without a uniform boundedness assumption.

2.3 Nonpreemptive Scheduling

The problem of nonpreemptive scheduling in the M/G/k has also seen attention, with results that complement those on preemptive scheduling. The most general such result is due to Glazebrook [8], who studies the discrete-time multiclass M/G/k with feedback. Each of the finitely many job classes may be of one of two types: preemptible with geometrically distributed service requirement, or nonpreemptible with generally distributed service requirement. There is a restriction on the latter type: letting S_i be the service requirement distribution of class i, the quantity $\mathbf{E}[S_i \mid S_i > s]$ must be uniformly bounded over all $s \geq 0$ and nonpreemptible classes i.

Glazebrook [8] shows that the Gittins policy is near-optimal under these conditions, specifically $\mathbf{E}[T_k] - \mathbf{E}[T_1] = O(k)$. Similarly to the previously discussed M/M/k system (Section 2.1.3), the strength of this result relies crucially on the assumption that $\mathbf{E}[S_i \mid S_i > s]$ is uniformly bounded, which in particular implies $\mathbf{Var}[S] < \infty$. One of the main challenges we overcome in proving Theorem 1.1 is bounding $\mathbf{E}[T_k] - \mathbf{E}[T_1]$ without a uniform boundedness assumption.

One can imagine a Markov process job model that allows for arbitrary state-based preemption constraints. Such a model includes preemptive, nonpreemptive, and partially preemptive settings, generalizing both our job model and that of Glazebrook [8]. Whether our techniques generalize to such a job model is an interesting open problem.

3 SYSTEM MODEL

We study preemptive scheduling in an M/G/k queue with arrival rate λ and service requirement distribution S. Each of the k servers has speed $1/k$, so the system has total service capacity 1 and load $\rho = \lambda\,\mathbf{E}[S]$, regardless of the number of servers. We assume $\rho < 1$ for stability.

We assume that preemption incurs no overhead. We also permit sharing of servers. For example, if $k = 2$ and there are three jobs in the system, the scheduler may choose to serve each job at rate $1/3$. More generally, at every moment in time, the scheduler assigns each job a service rate such that the total service rate is at most 1 and each individual job's service rate is at most $1/k$.

Throughout this work we compare a k-server system with a single-server system with the same total service capacity. We denote the response time distribution in a k-server system by T_k. In particular, T_1 is the response time in a single server system. We omit the subscript when the number of servers is clear from context.

Our goal is to schedule to minimize mean response time $\mathbf{E}[T_k]$. By Little's law, this is equivalent to minimizing $\mathbf{E}[N_k] = \lambda\,\mathbf{E}[T_k]$, the mean number of jobs in the system.

Depending on the setting, the scheduler might have perfect, zero, or partial information on each job's service requirement. To capture the wide range of possible information available to the scheduler and preemption constraints, we use a very general job model. In the rest of this section, we motivate what we want from a job model (Section 3.1), we describe our job model (Section 3.2) and we showcase its flexibility with several examples (Section 3.3).

3.1 What is a Job?

The job model encodes the process by which the scheduler gains information about a job's service requirement. We ensure that we can handle a highly general set of information-revealing scenarios, including four scenarios illustrated in Fig. 3.1. The simplest information scenarios are perfect information, where the scheduler knows each job's service requirement immediately upon the job's arrival to the system, and zero information, where the scheduler only knows the overall service requirement distribution, but not an individual job's service requirement.

One step beyond these scenarios, we can also handle the partial information scenario. In the partial information scenario, upon each job's arrival the scheduler is given a service requirement distribution for that job that is more specific than the overall service requirement distribution. For instance, the scheduler could be given a service requirement estimate where the joint distribution of true service requirement and the estimate is known. This model is the size-estimate model considered by Mitzenmacher [15], so our results apply to this job model.

However, we need not only consider scenarios where information is only revealed when the job arrives in the system. For instance, we can handle a scenario where a job is broken up into a series of several stages with known service requirement distributions, and the scheduler learns when each stage is completed. Even more generally, we can handle a scenario where the sequence of stages that each job consists of is not known to the scheduler upon arrival, and is only revealed to the scheduler as the job is run and its stages complete. An example of such a job is shown on the right side of Fig. 3.1. This is the "multistage job model" considered by Scully et al. [19], so our results apply to that setting.

We unify these disparate scenarios and far more by encapsulating the information known to the scheduler into the "state" of a job, and allowing the state to evolve according to an arbitrary Markov process. We give the technical details in Section 3.2, and demonstrate how our model can handle the above scenarios in Section 3.3.

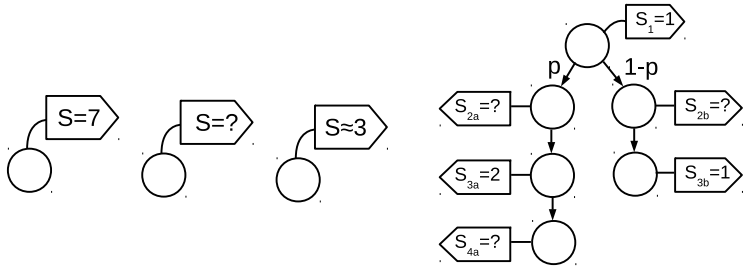

Fig. 3.1. Four job information scenarios. From left to right: a job with perfect job-size information; a job with zero job-size information; a job with partial job-size information; and a multistage job where some stages have perfect information, some stages have zero information, and the first stage transition is probabilistic.

3.2 Jobs are Absorbing Markov Processes

We model jobs as continuous-time absorbing Markov processes. The intuition is that a job's state encodes everything the scheduler knows about the job. A job's state evolves stochastically as it is served. The evolution of the state is not dependent on any decisions by the scheduler, beyond choosing whether to run the job or not.

3.2.1 Job State Space. Each job follows an i.i.d. Markovian trajectory $\{X(s)\}_{s\geq 0}$ through some *job state space* $\mathbb{X}$. Here s denotes the total service a job has received so far, so $X(s)$ is the random state of a job after service s. While a job is being served at rate $\dot{s}$, the job's state evolves at rate $\dot{s}$ according to the dynamics of the Markov process. In particular, a job's state does not change if it is not in service. More formally, if a job arrives at time 0 and receives service at rate $\dot{s}(t)$ at time t, then at time t, it is in state

$$X\left(\int_0^t \dot{s}(u)\,\mathrm{d}u\right).$$

3.2.2 Arrival and Departure States. Each arriving job enters the system in an i.i.d. *arrival state* $X_{\mathrm{A}} = X(0)$, which is the random initial state of the job's trajectory through the state space. A job completes and exits the system when it enters a designated *departure state* $\top \in \mathbb{X}$, which is the unique absorbing state. A job's service requirement, which we assume to have finite expectation, is the amount of service it needs to reach the departure state:

$$S \coloneqq \min\{s \geq 0 \mid X(s) = \top\}.$$

3.2.3 Technical Considerations. To keep our presentation maximally general, we have not made any explicit technical assumptions on the job Markov process. We discuss the conditions we implicitly assume in Appendix D. Roughly speaking, all we require is that the job Markov process be well behaved enough to define the Gittins game (Section 5).

3.3 The Markov Process Job Model is Extremely Flexible

Example 3.1 (Perfect Information). Consider a system where the scheduler knows each job's exact service requirement and preemption is allowed at any time. To model this, we let a job's state be its remaining service requirement. The job state space is $\mathbb{X} = \mathbb{R}_{\geq 0}$ with departure state $\top = 0$. A job's trajectory is $X(s) = (S - s)^+$. Although the arrival state $X_{\mathrm{A}} = S$ is stochastic, the dynamics are deterministic: a job's state decreases at rate 1 until it reaches 0.

Example 3.2 (Zero Information). Consider a system where the scheduler has zero a priori knowledge of any individual job's service requirement, but the the overall service requirement distribution S is known. To model this, we let a job's state be its *attained service*, which is the amount of service it has received so far. The state space is $\mathbb{X} = \mathbb{R}_{\geq 0} \cup \{\top\}$, and a job's trajectory is

$$X(s) = \begin{cases} s & \text{if } s < S \\ \top & \text{if } s \geq S. \end{cases}$$

Here the arrival state $X_\mathsf{A} = 0$ is deterministic, but the dynamics are stochastic: a job's state increases at rate 1 with stochastic jumps to the departure state $\top$, where the intensity of the jump process is the hazard rate of the service requirement distribution S.

Example 3.3 (Service Requirement Estimates). Consider a system where the scheduler receives an estimate of each job's service requirement which is correlated with its true service requirement. Specifically, suppose that each job announces to the scheduler an i.i.d. estimated service requirement M, and suppose the conditional distribution $(S \mid M)$ is known. To model this, we let a job's state be a tuple of its service requirement estimate and the amount of service it has received so far. The state space is $\mathbb{X} = \mathbb{R}^2_{\geq 0} \cup \{\top\}$, and a job's trajectory is

$$X(s) = \begin{cases} (M, s) & \text{if } s < S \\ \top & \text{if } s \geq S. \end{cases}$$

Here the arrival state $X_\mathsf{A} = (M, 0)$ is stochastic, as are the dynamics: the second element of a job's state increases at rate 1, and there are stochastic jumps to the departure state $\top$ according to the hazard rate of $(S \mid M)$.

We note that although we have focused on the case where M is a job's estimated service requirement, we can use the same technique to model any scenario in which jobs are "tagged" with static information. For example, in a system with multiple job classes, M could be a job's class.

Example 3.4 (Multistage Jobs). Consider a system where a job its broken up into stages, and upon completing stage i moves to stage j with probability p_{ij}, or else completes. Suppose the scheduler knows when a job completes a stage, and what stage the job transitions to. Furthermore, suppose the scheduler has zero knowledge of the length of each stage, but for each stage i the distribution S_i of the service requirement of the stage is known, as well as the transition matrix $\{p_{ij}\}$. To model this, we let a job's state be a tuple of its current stage and the amount of service it has received during this stage. We also specify that the jobs start in stage 0 upon arrival, and there are m possible stages. The state space is $\mathbb{X} = ([m] \times \mathbb{R}_{\geq 0}) \cup \{\top\}$. A job's trajectory depends on the sequence of stages that are sampled. For instance, if a job goes through stages 0, 3, and 5 and then departs, spending $s_0 \leftarrow S_0$, $s_3 \leftarrow S_3$, and $s_5 \leftarrow S_5$ time in each stage, then the job's trajectory would be

$$X(s) = \begin{cases} (0, s) & \text{if } s < s_0 \\ (3, s - s_0) & \text{if } s \in [s_0, s_0 + s_3) \\ (5, s - (s_0 + s_3)) & \text{if } s \in [s_0 + s_3, s_0 + s_3 + s_5) \\ \top & \text{if } s \geq s_0 + s_3 + s_5. \end{cases}$$

Here the arrival state $X_\mathsf{A} = (0, 0)$ is deterministic, but the dynamics are stochastic: the second element of a job's state increases at rate 1 with stochastic jumps to states of the form $(j, 0)$ or $\top$. The jump intensity is determined by the hazard rate of the current stage's service requirement distribution S_i, and the endpoint of the jump is sampled based on the transition matrix $\{p_{ij}\}$.

Although we have focused on the case where the service requirement distribution of each stage is unknown, we can use the same technique to model any mixture of known, unknown, and partially known distributions.

3.4 System State

We have thus far described the job state space. The state of the entire system is an infinite sequence of job states $(X_1, X_2, \ldots)$, where X_i is the random state of job i. Unless otherwise specified, we consider the equilibrium distribution of the system state.

In the context of the system state, the departure state $\top$ denotes the absence of a job. Because the system almost surely has finitely many jobs, we have $X_i = \top$ for all but finitely many i. We say a state x is *present* if $x \neq \top$.

In principle, jobs can be listed in any order in the system state $(X_1, X_2, \ldots)$. We use the convention that $(X_1, \ldots, X_k)$ are the jobs at the k servers. For most of the arguments in this work, the ordering $(X_1, \ldots, X_k)$ does not matter. We occasionally use one of the following conventions when it makes an argument easier to follow.

- In Lemma 8.1, we suppose that jobs are assigned to servers randomly. This makes the joint equilibrium distribution of $(X_1, \ldots, X_k)$ invariant under permutation.
- In Lemma 8.2, we suppose that present jobs are listed first. This ensures $X_k = \top$ if there are any idle servers.

4 THE GITTINS POLICY

The *Gittins* policy minimizes mean response time in the M/G/1 queue. It works in a way that easily generalizes to the M/G/k queue. The overall idea of Gittins is to assign each job a numerical priority, called its *Gittins rank* or simply "rank" (Definition 4.1), and preemptively prioritize jobs with lower rank.[2] In the M/G/1, this means serving the job of lowest rank, and in the M/G/k, this means serving the k jobs with the k lowest ranks.

Before we define the Gittins policy in full generality, we define it for the zero-information case (Example 3.2), where it takes a relatively simple form [1, 2]. In the zero-information case, a job's state is its attained service s, which is the amount of service the job has received so far, so a job's Gittins rank is a function of its attained service s:

$$\mathsf{rank}(s) = \inf_{t>s} \frac{\mathbf{E}[\min\{S,t\} - s \mid S > s]}{\mathbf{P}[S \leq t \mid S > s]}. \tag{4.1}$$

Figure 4.1 illustrates an example of the Gittins rank function in the zero-information case.

Intuitively, the numerator of (4.1) is the expected time for a job of attained service s to either complete or reach attained service t. The denominator is the probability that a job of attained service s will complete before it reaches attained service t. The infimum is the optimal time-per-completion ratio that can be achieved by running a job of attained service s for some amount of time $t - s$. The Gittins policy always serves the job of lowest rank, achieving the maximum completion rate across all jobs.

The Gittins rank function in the general case is defined in much the same way as (4.1). The primary difference is that instead of optimizing over a single stopping state t, we have to optimize over a *stopping set* $\mathbb{Y} \subseteq \mathbb{X}$, where $\mathbb{X}$ is the job state space.

[2]The lower-rank-is-better convention mirrors that of studies of Gittins in queueing and related optimization problems [6, 19, 20]. Other studies define a *Gittins index* and use a higher-index-is-better convention [1, 2, 7]. The Gittins rank and Gittins index are reciprocals of each other, so these two conventions are equivalent. We use the Gittins rank because it offers slightly better intuition in our setting.

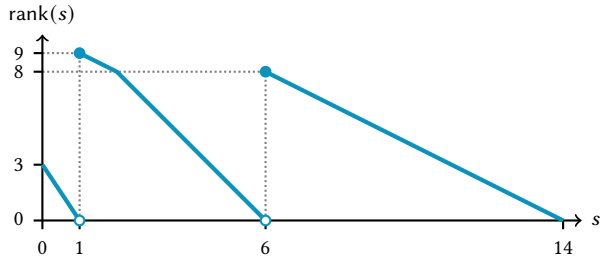

Fig. 4.1. Example of the Gittins rank function for the zero-information case, where a job's state is its attained service s. Each job's service requirement S is chosen uniformly from $\{1, 6, 14\}$.

Definition 4.1. The *Gittins rank function* $\mathsf{rank} : \mathbb{X} \to \mathbb{R}_{\geq 0}$ assigns to each state x a *Gittins rank*:[3]

$$\mathsf{rank}(x) \coloneqq \inf_{\mathbb{Y} \subseteq \mathbb{X}} \frac{\mathsf{service}(x, \mathbb{Y})}{\mathsf{done}(x, \mathbb{Y})},$$

where[4]

$$\begin{aligned}
\mathsf{service}(x, \mathbb{Y}) &\coloneqq \mathbf{E}[\text{service needed to first enter } \mathbb{Y} \mid \text{start from } x] \\
&= \mathbf{E}[\inf\{s \geq 0 \mid X(s) \in \mathbb{Y}\} \mid X(0) = x], \\
\mathsf{done}(x, \mathbb{Y}) &\coloneqq \mathbf{P}[\text{first state of } \mathbb{Y} \text{ entered is } \top \mid \text{start from } x] \\
&= \mathbf{P}[X(\inf\{s \geq 0 \mid X(s) \in \mathbb{Y}\}) = \top \mid X(0) = x].
\end{aligned}$$

As a corner case, we define $\mathsf{rank}(\top) \coloneqq \infty$.

In the zero-information case, Definition 4.1 is equivalent to (4.1). Specifically, choosing stopping set $\{t, \top\}$ in Definition 4.1 corresponds to choosing t in (4.1).[5]

Gittins breaks rank ties arbitrarily. If the jobs tied for lowest rank all have rank that increases with service, prioritizing by lowest rank naturally leads to sharing servers, as is typical of Foreground-Background and related scheduling policies [20, Appendix B].

5 THE GITTINS GAME

This section presents an alternative way of characterizing the Gittins rank function. It is based on an optimization problem which we call the *Gittins game*. While variations on the Gittins game have been used in past studies of Gittins [6, 7, 19, 23], we make a two-fold contribution. First, we define and study the Gittins game in a more general setting, namely that of our Markov process job model. Second, and more importantly, we draw new connections between properties of the Gittins game and mean response time throughout Sections 6–8. These new connections are crucial to proving our main results.

5.1 How to Play the Gittins Game

In the Gittins game, we are given a job in some initial state $x \in \mathbb{X}$ and a *penalty* $r \geq 0$. We serve the job at rate 1 for as long as we like, observing its state as it evolves during service. We may choose to stop at any time, which ends the game. When we stop, if the job is incomplete, meaning its state

[3] We restrict the infimum to stopping sets $\mathbb{Y} \subseteq \mathbb{X}$ such that $\mathsf{done}(x, \mathbb{Y}) > 0$.

[4] Strictly speaking, Section 3.2 defines $\{X(s)\}_{s \geq 0}$ to be the random trajectory of a job, but it may be that jobs never arrive in state $X(0) = x$. By conditioning on $X(0) = x$, we mean that $\{X(s)\}_{s \geq 0}$ is a random trajectory starting at x and following the dynamics of the job Markov process. We use this slight abuse of the notation throughout the paper.

[5] More generally, in the zero-information case, stopping sets are subsets of $\mathbb{R}_{\geq 0} \cup \{\top\}$ (see Example 3.2), and choosing $\mathbb{Y}$ in Definition 4.1 corresponds to choosing $t = \inf(\mathbb{Y} \setminus (\{\top\} \cup [0, s]))$ in (4.1).

is not $\top$, we pay the penalty r. The goal of the Gittins game is to minimize the expected sum of the time spent serving the job plus the penalty paid.

Because jobs are Markov processes, the solution to the Gittins game has the following form: choose a stopping set $\mathbb{Y} \subseteq \mathbb{X}$ and stop serving the job the first time its state enters $\mathbb{Y}$.[6] Recalling Definition 4.1 and letting

$$\mathsf{undone}(x, \mathbb{Y}) \coloneqq 1 - \mathsf{done}(x, \mathbb{Y}),$$

the expected cost of serving the job up until it enters $\mathbb{Y}$ is

$$\mathsf{game}(x, r, \mathbb{Y}) \coloneqq \mathsf{service}(x, \mathbb{Y}) + r\,\mathsf{undone}(x, \mathbb{Y}). \tag{5.1}$$

Minimizing the cost of the Gittins game amounts to choosing the best stopping set $\mathbb{Y}$. Because jobs are Markov processes, one can choose a stopping set that is simultaneously optimal for all initial states x.

Definition 5.1. The *optimal cost* of the Gittins game with initial state x and penalty r is

$$\mathsf{game}(x, r) \coloneqq \inf_{\mathbb{Y}} \mathsf{game}(x, r, \mathbb{Y}).$$

The *optimal stopping set* for penalty r is

$$\mathbb{Y}^*(r) \coloneqq \{x \in \mathbb{X} \mid \mathsf{game}(x, r) = r\}.$$

As suggested by the names, it is indeed the case that[7]

$$\mathsf{game}(x, r) = \mathsf{game}(x, r, \mathbb{Y}^*(r)).$$

As a shorthand, we write $\mathsf{service}(x, r) \coloneqq \mathsf{service}(x, \mathbb{Y}^*(r))$, and similarly for done and undone. As a corner case, we define $\mathsf{service}(\top, r) \coloneqq 0$ and $\mathsf{undone}(\top, r) \coloneqq 1$, so $\mathsf{game}(\top, r) = r$.

5.2 Properties of the Gittins Game

We now prove some properties of the Gittins game which will help us relate the Gittins game to the Gittins rank function. Some of the properties are also important in our mean response time analysis. Our first property is a general bound on $\mathsf{game}(x, r)$.

Lemma 5.2. *For all $x \in \mathbb{X}$ and $r \geq 0$,*

$$\mathsf{service}(x, r) \leq \mathsf{game}(x, r) \leq \min\{r, \mathsf{service}(x, \infty)\}.$$

Proof. Definition 5.1 implies $\mathsf{service}(x, r) \leq \mathsf{game}(x, r)$. The upper bound on $\mathsf{game}(x, r)$ follows from stopping as early or as late as possible in the Gittins game. On one hand, we could stop immediately and pay the penalty r, so $\mathsf{game}(x, r) \leq \mathsf{game}(x, r, \mathbb{X}) = r$. On the other hand, we could serve the job until it completes and avoid the penalty, so $\mathsf{game}(x, r) \leq \mathsf{game}(x, r, \{\top\}) = \mathsf{service}(x, \{\top\})$. If $r = \infty$, then we must avoid paying the penalty, so $\mathsf{service}(x, \{\top\}) = \mathsf{service}(x, \infty)$. □

Our next property characterizes the derivative $\frac{\mathrm{d}}{\mathrm{d}r}\,\mathsf{game}(x, r)$. For any fixed $\mathbb{Y}$, by (5.1),

$$\frac{\mathrm{d}}{\mathrm{d}r}\,\mathsf{game}(x, r, \mathbb{Y}) = \mathsf{undone}(x, \mathbb{Y}). \tag{5.2}$$

The intuition is that increasing r increases our cost when we pay the penalty, which happens with probability $\mathsf{undone}(x, \mathbb{Y})$. The following lemma, proven in Appendix B.1, shows that the same intuition works for the optimal cost, even though the optimal stopping set varies with r.

[6] As we discuss in Appendix D, this conclusion relies on some technical assumptions on the job model.

[7] To see why this is true, notice that if $\mathsf{game}(x, r) = r$, then stopping and paying the penalty r is an optimal action when in state x. For further details, see Appendix D.

Lemma 5.3. *For all $x \in \mathbb{X}$, the function $r \mapsto \mathsf{game}(x, r)$ is continuous, concave, and differentiable almost everywhere with*

$$\frac{\mathrm{d}}{\mathrm{d}r}\,\mathsf{game}(x, r) = \mathsf{undone}(x, r).$$

5.3 Relationship to the Gittins Rank Function

We now discuss how the Gittins game relates to the Gittins rank function. Recall from Section 4 that $\mathsf{rank}(x)$ is the smallest time-per-completion ratio $\mathsf{service}(x, \mathbb{Y})/\mathsf{done}(x, \mathbb{Y})$ that we can achieve by optimizing the stopping set $\mathbb{Y}$.

Lemma 5.4. *For all $x \in \mathbb{X}$ and $r \geq 0$,*

$$\mathbb{Y}^*(r) = \{x \in \mathbb{X} \mid \mathsf{rank}(x) \geq r\}.$$

Proof. We can rewrite $\mathsf{game}(x, r, \mathbb{Y})$ in terms of the time-per-completion ratio:

$$\mathsf{game}(x, r, \mathbb{Y}) = r - \mathsf{done}(x, \mathbb{Y})\left(r - \frac{\mathsf{service}(x, \mathbb{Y})}{\mathsf{done}(x, \mathbb{Y})}\right). \tag{5.3}$$

If there exists $\mathsf{service}(x, \mathbb{Y})/\mathsf{done}(x, \mathbb{Y}) < r$ for some $\mathbb{Y} \subseteq \mathbb{X}$, then (5.3) implies $\mathsf{game}(x, r, \mathbb{Y}) < r$, so it is strictly suboptimal to stop in state x. If instead $\mathsf{service}(x, \mathbb{Y})/\mathsf{done}(x, \mathbb{Y}) \geq r$ for all $\mathbb{Y} \subseteq \mathbb{X}$, then (5.3) implies $\mathsf{game}(x, r, \mathbb{Y}) \geq r$ for all $\mathbb{Y} \subseteq \mathbb{X}$, so it is optimal to stop in state x. □

Corollary 5.5. *For all $x \in \mathbb{X}$ and $0 \leq r \leq \mathsf{rank}(x)$, because $x \in \mathbb{Y}^*(r)$, we have $\mathsf{service}(x, r) = 0$.*

6 COMPUTING MEAN RESPONSE TIME VIA RELEVANT WORK

One of the primary methods of analyzing the mean response time in the M/G/1 is the "tagged job" method. The tagged job method has been successfuly applied to a wide variety of scheduling policies in the M/G/1 [11, 12, 20]. Recent works have used the tagged job method in the M/G/k, but it applies to only certain classes of scheduling policies [10, 18]. In particular, it has not been successfully applied to Gittins, and there are several obstacles to doing so [18, Appendix A].

In light of the difficulty of applying the tagged job method in the M/G/k, we introduce a new method of computing mean response time. Our method is based on the Gittins rank function and the Gittins game, so it is best suited to analyzing the mean response time of Gittins, but could in principle work for any scheduling policy. We give some intuition for our technique (Section 6.1) before using it to derive the mean response time formula (Section 6.2).

6.1 Thinking in Terms of Work

To compute mean response time $\mathbf{E}[T]$, it suffices to compute the mean number of jobs in the system $\mathbf{E}[N]$ and apply Little's law. This begs the question: how do we compute $\mathbf{E}[N]$? To build intuition, we start our discussion with the M/G/1, but our technique will apply equally well to the M/G/k.

When using a complex scheduling policy, the simplest quantity to compute in the M/G/1 is neither $\mathbf{E}[T]$ nor $\mathbf{E}[N]$ but rather $\mathbf{E}[W]$, the mean amount of work, where *work* W is the total remaining service requirement of all jobs in the system. The reason $\mathbf{E}[W]$ is so easy to compute is that in the M/G/1, $\mathbf{E}[W]$ does not depend on the scheduling policy, assuming the policy is work-conserving.

Given that $\mathbf{E}[W]$ is easy to compute, we might ask: can we use $\mathbf{E}[W]$ to compute $\mathbf{E}[N]$? There is a system where the answer is yes: the M/M/1. Specifically, if the service rate is μ, every job in the system has expected remaining service requirement $1/\mu$, so $\mathbf{E}[N] = \mu\,\mathbf{E}[W]$. In fact, this relationship depends only on the service requirement distribution, so it holds in the M/M/k as well.

Our approach is to generalize the $\mathbf{E}[N] = \mu\,\mathbf{E}[W]$ relationship to handle non-exponential service requirement distributions. To do so, we use the Gittins rank function to make a new connection between $\mathbf{E}[N]$ and the amount of "relevant" work in the system (Theorem 6.3).

6.2 A New Formula for Mean Response Time

6.2.1 Relevant States, Jobs, and Work. We call a state x *r-relevant*, or simply "relevant" when r is clear from context, if $\mathsf{rank}(x) < r$. Similarly, we call a job r-relevant if its state is r-relevant. A job or state that is not r-relevant is *r-irrelevant*, or simply "irrelevant".

To generalize the $\mathbf{E}[N] = \mu\,\mathbf{E}[W]$ relationship from Section 6.1, we focus on work contributed by r-relevant jobs. The twist is that our formula for $\mathbf{E}[N]$ integrates the amount of work contributed by r-relevant jobs over all values of r.

The first step in deriving the mean response time formula is to formalize what we mean by work contributed by r-relevant jobs.

Definition 6.1. The *r-relevant remaining service requirement* of a job in state x is the amount of service the job needs to reach an r-irrelevant state:

$$S(x, r) \coloneqq \Big[\inf\{s \geq 0 \mid \mathsf{rank}(X(s)) \geq r\} \;\Big|\; X(0) = x\Big].$$

The *r-relevant work* in the system is the total of all jobs' r-relevant remaining service requirements:

$$W(r) \coloneqq \sum_{i=1}^{\infty} S(X_i, r).$$

The intuition is that $S(x, r)$ is the amount of service a job in state x needs to either complete or reach rank r. Recalling Definition 5.1, we see

$$\mathbf{E}[S(x, r)] = \mathsf{service}(x, r). \tag{6.1}$$

6.2.2 Counting Jobs via Relevant Work. We now express the mean number of jobs $\mathbf{E}[N]$ in terms of relevant work. The first step is the following lemma, which "counts" whether a single job is present using the job's relevant remaining service requirement.

Lemma 6.2. *For all $x \in \mathbb{X}$,*

$$\int_0^\infty \frac{\mathsf{service}(x, r)}{r^2}\,\mathrm{d}r = \mathbb{1}(x \neq \top).$$

Proof. By Definition 5.1, we have $\mathsf{service}(\top, r) = 0$ for all $r \geq 0$, so only the $x \neq \top$ case remains. Lemma 5.3 and Definition 5.1 imply

$$\frac{\mathrm{d}}{\mathrm{d}r}\frac{\mathsf{game}(x, r)}{r} = \frac{r\frac{\mathrm{d}}{\mathrm{d}r}\mathsf{game}(x, r) - \mathsf{game}(x, r)}{r^2} = \frac{-\mathsf{service}(x, r)}{r^2},$$

so by Lemmas 5.2 and 5.3,

$$\int_0^\infty \frac{\mathsf{service}(x, r)}{r^2}\,\mathrm{d}r = \lim_{r\to 0}\frac{\mathsf{game}(x, r)}{r} - \lim_{r\to\infty}\frac{\mathsf{game}(x, r)}{r} = 1 - 0. \qquad \square$$

Because relevant work is the sum of each job's relevant remaining service requirement, the same type of integral as in Lemma 6.2 but with $\mathbf{E}[W(r)]$ instead of $\mathsf{service}(x, r)$ counts the expected number of jobs present, which is $\mathbf{E}[N] = \lambda\,\mathbf{E}[T]$.

Theorem 6.3. *The mean number of jobs and mean response time in an M/G/k under any scheduling policy can be expressed in terms of mean relevant work:*

$$\mathbf{E}[N] = \lambda\,\mathbf{E}[T] = \int_0^\infty \frac{\mathbf{E}[W(r)]}{r^2}\,\mathrm{d}r.$$

7 RELEVANT WORK DECOMPOSITION LAW

Theorem 6.3 shows us how to compute mean response time in terms of mean relevant work. This means that if we want to analyze Gittins's mean response time gap, $\mathbf{E}[T_k] - \mathbf{E}[T_1]$, it suffices to analyze its mean r-relevant work gap, $\mathbf{E}[W_k(r)] - \mathbf{E}[W_1(r)]$. Put another way, we are looking for a "work decomposition law" that decomposes $\mathbf{E}[W_k(r)]$ into $\mathbf{E}[W_1(r)]$ plus another term.

Work decomposition laws have been derived for a wide variety of queueing systems. However, these prior results are limited systems with finitely many job classes and either nonpreemptible jobs [3, 4, 8] or exponential service requirements [9]. What these systems have in common is that every job in the queue is in one of finitely many states. Proving work decomposition laws in such systems involves reasoning about the number of jobs in each state. Unfortunately, such an approach would be very complicated in our model, because the job state space may be uncountable.[8]

Our key idea is to approach analyzing $\mathbf{E}[W_k(r)]$ by reasoning directly in terms of relevant work. This results in a very general relevant work decomposition law whose proof is much simpler than proofs of similar prior results. As a warm-up, we first prove a decomposition law for total work (Section 7.1) before tackling the more complicated case of relevant work (Sections 7.2 and 7.3).

7.1 Warm-up: Total Work Decomposition Law

Consider the M/G/k under any scheduling policy. Our goal in this warm-up is to characterize the mean total work in the system $\mathbf{E}[W_k]$. To clarify, the work in the system is the sum of all jobs' remaining service requirements: $W_k \coloneqq \sum_{i=1}^{\infty} S(X_i, \infty)$.

7.1.1 The Rate Conservation Law. The total amount of work W_k fluctuates over time. But in an equilibrium system, the distribution of W_k remains constant over time. Therefore, the average rate at which W_k increases equals the average rate at which W_k decreases, an instance of the *rate conservation law* from Palm calculus [16].

We now apply the rate conservation law to W_k. This turns out not to be directly useful, but we walk through the process as an intuition-building exercise.

- Decreases in W_k occur continuously as jobs receive service. At any moment in time, the rate of decrease is the current service rate, which is

$$B_k \coloneqq \text{fraction of servers busy} = \frac{1}{k} \sum_{i=1}^{k} \mathbb{1}(X_i \neq \top),$$

  so the average rate at which W_k decreases is $\mathbf{E}[B_k]$.
- Increases in W_k are jumps caused by new arrivals. Arrivals occur at average rate λ and have average service requirement $\mathbf{E}[S]$, so the average rate at which W_k increases is $\lambda\,\mathbf{E}[S] = \rho$.

Equating the average increase and decrease rates yields an unsurprising result: $\mathbf{E}[B_k] = \rho$.

7.1.2 Deriving Work Decomposition from Rate Conservation. The key to characterizing $\mathbf{E}[W_k]$ is to apply the rate conservation law to W_k^2 instead of W_k.

- Because W_k decreases at rate B_k, the average rate at which W_k^2 decreases is $2\,\mathbf{E}[B_k W_k]$. Although B_k and W_k are not independent, we can write this as $2(\mathbf{E}[W_k] - \mathbf{E}[(1 - B_k)W_k])$.
- Because Poisson arrivals see time averages [24], the average rate at which W_k^2 increases is $\lambda\,\mathbf{E}[(S + W_k)^2 - W_k^2] = \lambda\,\mathbf{E}[S^2 + 2SW_k]$. Because the service requirement of an arriving job is independent of the amount of work in the system when it arrives, we can rewrite this as $\lambda\,\mathbf{E}[S^2] + 2\rho\,\mathbf{E}[W_k]$.

[8] Recent studies have derived *worst-case* bounds on the mean relevant work gap between the M/G/k and M/G/1 for certain scheduling policies [10, 18]. However, when applied to Gittins, this worst-case approach yields bounds that are too weak.

Equating the average increase and decrease rates and solving for $\mathbf{E}[W_k]$ yields

$$\mathbf{E}[W_k] = \frac{\frac{\lambda}{2}\mathbf{E}[S^2] + \mathbf{E}[(1-B_k)W_k]}{1-\rho}.$$

When $k = 1$, the server is busy if and only if there is work in the system: $B_1 = \mathbb{1}(W_1 > 0)$. This means $\mathbf{E}[(1-B_1)W_1] = 0$, which implies the following decomposition law characterizing the mean work gap between the M/G/k and M/G/1:

$$\mathbf{E}[W_k] - \mathbf{E}[W_1] = \frac{\mathbf{E}[(1-B_k)W_k]}{1-\rho}. \tag{7.1}$$

7.1.3 Interpreting the Work Decomposition Law. The mean work gap given by the right-hand side of (7.1) prompts two questions. First, is there an intuitive interpretation of the gap? Second, is there any hope of bounding the gap?

To intuitively interpret the gap, note that $1 - B_k$ is only nonzero if there is an idle server. This means there are at most $k-1$ jobs contributing to the work W_k in the $\mathbf{E}[(1-B_k)W_k]$ term. Because $\mathbf{E}[1-B_k] = 1-\rho$, we can think of (7.1) as very roughly saying that the mean work in an M/G/k is the mean work in an M/G/1 plus the work of at most $k-1$ jobs.

This interpretation gives us a clue as to how we might bound the gap. Suppose there exists an $s_{\max} \geq 0$ such that for every state $x \in \mathbb{X}$, we can bound the expected remaining service requirement starting at x as $\text{service}(x, \infty) \leq s_{\max}$. One can use this bound to formalize the above intuition, showing that the right-hand side of (7.1) is at most $(k-1)s_{\max}$.

7.1.4 Why Relevant Work is Easier to Bound than Total Work. In general, there is no uniform bound $s_{\max}$ on a job's expected total remaining service requirement. However, if we shift our focus to *r-relevant* work, we do have such a bound: Lemma 5.2 implies $\text{service}(x, r) \leq r$. This means that if we can prove an analogue of (7.1) for relevant work, then there is hope of bounding the resulting mean relevant work gap. We take exactly this approach: the remainder of this section generalizes (7.1) to relevant work, and Section 8 bounds the resulting gap.

7.2 Fresh and Recycled Relevant Jobs

Just as we proved (7.1) by applying the rate conservation law to W_k^2, we will prove an analogous result for relevant work by applying the rate conservation law to $W_k(r)^2$. In order to do this, we need to understand how $W_k(r)$ increases and decreases. The way $W_k(r)$ decreases is by serving r-relevant jobs, which happens at rate

$$B_k(r) \coloneqq \text{fraction of servers busy with } r\text{-relevant jobs} = \frac{1}{k}\sum_{i=1}^{k} \mathbb{1}(\text{rank}(X_i) < r).$$

There are two ways $W_k(r)$ increases:

- New arrivals can enter the system with a relevant arrival state. We call such jobs *r-fresh*.
- Jobs that are already in the system can transition from an irrelevant state to a relevant state. We call the moments when this happens *r-recyclings*, and we call such jobs *r-recycled*.

As with "relevant", we sometimes omit the prefix r. In order to state the relevant work decomposition law, we need some additional definitions and notation related to fresh and recycled jobs.

We start by clarifying that whether a job is fresh or recycled varies over time. Consider an arbitrary relevance threshold $r \geq 0$. We classify each job into one of three categories based on its current and past states:[9]

[9]Scully et al. [20] use the same categorization with slightly different terminology when analyzing M/G/1 response time under a wide range of scheduling policies.

- r-irrelevant;
- r-fresh, meaning it is currently r-relevant and has always been r-relevant; or
- r-recycled, meaning it is currently r-relevant but was r-irrelevant at some point in the past.

Every job arrives as either r-irrelevant or r-fresh and then alternates between r-irrelevant and r-recycled until its completion.

7.2.1 Notation for Fresh and Recycled Jobs. Fresh jobs appear due to arrivals, which we already have notation for (Section 3): arrivals occur at average rate λ; the random state of a job that just arrived is X_A; and Poisson arrivals see time averages [24], so arrivals see the equilibrium distribution of the system state $(X_1, X_2, \ldots)$, as well as quantities like $W_k(r)$ which are functions of the system state. We now define analogous notation for recycled jobs:

- The average rate of r-recyclings is
$$\lambda_\mathrm{R}(r) \coloneqq \lambda\, \mathbf{E}[\text{number of } r\text{-recyclings during a job's trajectory}].$$
- We write $X_\mathrm{R}(r)$ to denote the random state of a job just *after* it becomes r-recycled.
- Unlike Poisson arrivals, r-recyclings do not necessarily see an equilibrium distribution of the system state. We write $\mathbf{E}_r[\cdot]$ to denote expectation with respect to the system state sampled at moments just *before* r-recyclings.

The following definition gives an example of the above notation.

Definition 7.1. The *r-fresh load* is the average rate at which relevant work increases due to fresh jobs arriving:
$$\rho_\mathrm{A}(r) \coloneqq \lambda\, \mathbf{E}[S(X_\mathrm{A}, r)].$$
The *r-recycled load* is the average rate at which relevant work increases due to recyclings:
$$\rho_\mathrm{R}(r) \coloneqq \lambda_\mathrm{R}(r)\, \mathbf{E}_r[S(X_\mathrm{R}(r), r)].$$

For ease of presentation, we have implicitly assumed $\lambda_\mathrm{R}(r) < \infty$. We discuss how to relax this assumption in Appendix E.

7.3 Generalizing the Decomposition Law to Relevant Work

Having introduced notation for fresh and recycled jobs, we are now ready to state and prove a decomposition law for relevant work analogous to (7.1).

Theorem 7.2. *For all $r \geq 0$, the mean r-relevant work gap between the M/G/k and M/G/1 under Gittins is*
$$\mathbf{E}[W_k(r)] - \mathbf{E}[W_1(r)] = \frac{\mathbf{E}\big[(1 - B_k(r))\, W_k(r)\big] + \lambda_\mathrm{R}(r)\, \mathbf{E}_r\big[S(X_\mathrm{R}(r), r)\, W_k(r)\big]}{1 - \rho_\mathrm{A}(r)}.$$

Proof. We begin by deriving an expression for $\mathbf{E}[W_k(r)]$ that holds for any number of servers k. To reduce clutter, we drop the subscript unless specifically studying the $k = 1$ case.

To obtain an expression for $\mathbf{E}[W(r)]$, we apply the rate conservation law (Section 7.1.1) to $W(r)^2$.

- The average rate at which $W(r)^2$ decreases is $2\, \mathbf{E}[B(r)\, W(r)]$.
- The average rate at which $W(r)^2$ increases is

(increase rate due to fresh arrivals) + (increase rate due to r-recyclings)
$$= \lambda\, \mathbf{E}\big[(W(r) + S(X_\mathrm{A}, r))^2 - W(r)^2\big] + \lambda_\mathrm{R}(r)\, \mathbf{E}_r\big[(W(r) + S(X_\mathrm{R}(r), r))^2 - W(r)^2\big]$$
$$= \lambda\, \mathbf{E}[S(X_\mathrm{A}, r)^2] + \lambda_\mathrm{R}(r)\, \mathbf{E}_r[S(X_\mathrm{R}(r), r)^2] + 2\, \rho_\mathrm{A}(r)\, \mathbf{E}[W(r)] + 2\, \lambda_\mathrm{R}(r)\, \mathbf{E}_r\big[S(X_\mathrm{R}(r), r)\, W(r)\big].$$

Equating the decrease and increase rates, we find

$$\mathbf{E}[W(r)] = \frac{1}{1-\rho_{\mathrm{A}}(r)}\Bigg(\frac{\lambda}{2}\,\mathbf{E}[S(X_{\mathrm{A}},r)^2] + \frac{\lambda_{\mathrm{R}}(r)}{2}\,\mathbf{E}_r[S(X_{\mathrm{R}},r)^2] + \mathbf{E}\big[(1-B(r))\,W(r)\big] + \lambda_{\mathrm{R}}(r)\,\mathbf{E}_r\big[S(X_{\mathrm{R}}(r),r)\,W(r)\big]\Bigg). \tag{7.2}$$

To complete the proof, it suffices to show that if $k = 1$, then $\mathbf{E}\big[(1-B_1(r))\,W_1(r)\big] = 0$ and $\mathbf{E}_r\big[S(X_{\mathrm{R}}(r),r)\,W_1(r)\big] = 0$, as the other terms do not depend on k.

- At any moment in time, if there are any r-relevant jobs in the system, then an r-relevant job is in service. This means $B_1(r) = \mathbb{1}(W_1(r) > 0)$, and thus $\mathbf{E}\big[(1-B_1(r))\,W_1(r)\big] = 0$.
- Just before an r-recycling, the job that is about to be r-recycled is in service, so there are no r-relevant jobs in the system. This means $W_1(r) = 0$, so $\mathbf{E}_r\big[S(X_{\mathrm{R}}(r),r)\,W_1(r)\big] = 0$. □

7.4 Optimality of Gittins in the M/G/1

Combining Theorem 6.3 with the reasoning behind Theorem 7.2 yields an elegant proof of the optimality of Gittins in the M/G/1.

Theorem 7.3. *The Gittins policy minimizes mean response time in the M/G/1.*

Proof. The key observation is that (7.2) actually holds under *any* scheduling policy, not just Gittins. This means the conclusion of Theorem 7.2 is valid for an M/G/k under *any* scheduling policy, provided that the M/G/1 is under Gittins. In particular, taking $k = 1$ implies that for all $r \geq 0$,

$$\mathbf{E}[W_{\text{M/G/1 under any scheduling policy}}(r)] - \mathbf{E}[W_{\text{M/G/1 under Gittins}}(r)] \geq 0,$$

so the result follows from Theorem 6.3. □

8 PROOFS OF MAIN RESULTS

We are now ready to prove our main results. Armed with tools from Sections 5–7, there are two main steps to the proof. The first step is to bound the mean relevant work gap between the M/G/k and M/G/1 under Gittins (Section 8.1), which we do using properties of the Gittins game. The second step is to bound the integral of this mean work gap bound (Section 8.3), which yields a bound on the mean response time gap via Theorem 6.3. This second step requires some additional properties of the Gittins game (Section 8.2).

To reduce clutter, in this section we usually omit the subscript k from $W_k(r)$ and $B_k(r)$.

8.1 Bounding Relevant Work

The following two lemmas bound terms from the mean relevant work gap in Theorem 7.2.

Lemma 8.1. *For all $r \geq 0$,*

$$\mathbf{E}[(1-B(r))\,W(r)] \leq (k-1)\min\{\rho, 1-\rho\}\,\mathbf{E}[\text{service}(X_1,r) \mid X_1 \neq \top \text{ and } X_2 = \top] + (k-1)r\min\{\rho_{\mathrm{A}}(r) + \rho_{\mathrm{R}}(r), \rho - \rho_{\mathrm{A}}(r) - \rho_{\mathrm{R}}(r)\},$$

where the notation in the first term of the right-hand side uses the convention that we label jobs such that the equilibrium distribution of $(X_1,\ldots,X_k)$ is invariant under permutation.

Proof. We have $1 - B(r) = 1/k \cdot \sum_{j=1}^{k} \mathbb{1}(\mathsf{rank}(X_j) \geq r)$. In particular, whenever $1 - B(r) > 0$, fewer than k jobs are r-relevant, so

$$\begin{aligned}
&\mathbf{E}\big[(1 - B(r))\, W(r)\big] \\
&\quad = \mathbf{E}\left[\left(\frac{1}{k}\sum_{j=1}^{k} \mathbb{1}(\mathsf{rank}(X_j) \geq r)\right)\left(\sum_{i=1}^{k} \mathbf{E}[S(X_i, r) \mid X_1, \ldots, X_k]\right)\right] \\
&\quad = \frac{1}{k}\sum_{i=1}^{k}\sum_{j=1}^{k} \mathbb{1}(i \neq j)\, \mathbf{E}\big[\mathsf{service}(X_i, r)\, \mathbb{1}(\mathsf{rank}(X_j) \geq r)\big] \qquad \text{[by Corollary 5.5]} \\
&\quad = (k-1)\, \mathbf{E}\big[\mathsf{service}(X_1, r)\, \mathbb{1}(\mathsf{rank}(X_2) \geq r)\big],
\end{aligned}$$

where the last line uses the convention that the equilibrium distribution of $(X_1, \ldots, X_k)$ is invariant under permutation. Our approach is to split $\mathbf{E}[\mathsf{service}(X_1, r)\, \mathbb{1}(\mathsf{rank}(X_2) > r)]$ into two pieces

$$\begin{aligned}
\mathbf{E}\big[\mathsf{service}(X_1, r)\, \mathbb{1}(\mathsf{rank}(X_2) > r)\big] &= \mathbf{E}\big[\mathsf{service}(X_1, r)\, \mathbb{1}(X_1 \neq \top \text{ and } X_2 = \top)\big] \\
&\quad + \mathbf{E}\big[\mathsf{service}(X_1, r)\, \mathbb{1}(\mathsf{rank}(X_1) < r \leq \mathsf{rank}(X_2) < \infty)\big]
\end{aligned}$$

and bound each piece separately. Because

$$\mathbf{P}[X_1 \neq \top \text{ and } X_2 = \top] \leq \min\big\{\mathbf{P}[X_1 \neq \top], \mathbf{P}[X_2 = \top]\big\} \leq \min\{\rho, 1 - \rho\},$$

we can bound the first piece as

$$\mathbf{E}\big[\mathsf{service}(X_1, r)\, \mathbb{1}(X_1 \neq \top \text{ and } X_2 = \top)\big] \leq \min\{\rho, 1 - \rho\}\, \mathbf{E}\big[\mathsf{service}(X_1, r) \,\big|\, X_1 \neq \top \text{ and } X_2 = \top\big].$$

Similarly, because

$$\begin{aligned}
\mathbf{P}[\mathsf{rank}(X_1) < r \leq \mathsf{rank}(X_2) < \infty] &\leq \min\big\{\mathbf{P}[\mathsf{rank}(X_1) < r], \mathbf{P}[r \leq \mathsf{rank}(X_2) < \infty]\big\} \\
&\leq \min\{\rho_{\mathsf{A}}(r) + \rho_{\mathsf{R}}(r), \rho - \rho_{\mathsf{A}}(r) - \rho_{\mathsf{R}}(r)\},
\end{aligned}$$

we can bound the second piece as

$$\begin{aligned}
&\mathbf{E}\big[\mathsf{service}(X_1, r)\, \mathbb{1}(\mathsf{rank}(X_1) < r \leq \mathsf{rank}(X_2) < \infty)\big] \\
&\quad \leq r\, \mathbf{P}[\mathsf{rank}(X_1) < r \leq \mathsf{rank}(X_2) < \infty] \qquad \text{[by Lemma 5.2]} \\
&\quad \leq r \min\{\rho_{\mathsf{A}}(r) + \rho_{\mathsf{R}}(r), \rho - \rho_{\mathsf{A}}(r) - \rho_{\mathsf{R}}(r)\}.
\end{aligned}$$

□

Lemma 8.2. *For all $r \geq 0$, we have $\lambda_{\mathsf{R}}(r)\, \mathbf{E}_r\big[S(X_{\mathsf{R}}(r), r)\, W(r)\big] \leq (k-1) r\, \rho_{\mathsf{R}}(r)$.*

Proof. All r-recyclings happen while fewer than k jobs are r-relevant. Using the convention that we list present jobs first in the system state $(X_1, X_2, \ldots)$, we therefore have

$$\mathbf{E}_r\big[S(X_{\mathsf{R}}(r), r)\, W(r)\big] = \mathbf{E}_r\left[S(X_{\mathsf{R}}(r), r) \sum_{i=1}^{k-1} S(X_i, r)\right].$$

For all $i \in \{1, \ldots, k-1\}$, the random variables $S(X_{\mathsf{R}}(r), r)$ and $S(X_i, r)$ are conditionally independent given $X_{\mathsf{R}}(r)$ and X_i, so

$$\begin{aligned}
&\lambda_{\mathsf{R}}(r)\, \mathbf{E}_r\big[S(X_{\mathsf{R}}(r), r)\, S(X_i, r)\big] \\
&\quad = \lambda_{\mathsf{R}}(r)\, \mathbf{E}_r\big[\mathbf{E}[S(X_{\mathsf{R}}(r), r) \mid X_{\mathsf{R}}(r), X_i]\, \mathbf{E}[S(X_i, r) \mid X_{\mathsf{R}}(r), X_i]\big] \\
&\quad = \lambda_{\mathsf{R}}(r)\, \mathbf{E}_r\big[\mathbf{E}[S(X_{\mathsf{R}}(r), r) \mid X_{\mathsf{R}}(r), X_i]\, \mathsf{service}(X_i, r)\big] \\
&\quad \leq r\, \lambda_{\mathsf{R}}(r)\, \mathbf{E}_r[S(X_{\mathsf{R}}(r), r)] \qquad \text{[by Lemma 5.2]} \\
&\quad = r\, \rho_{\mathsf{R}}(r). \qquad \text{[by Definition 7.1]}
\end{aligned}$$

□

8.2 The Gittins Game from the Arrival State

Combining Lemmas 8.1 and 8.2 bounds the mean r-relevant work gap in terms of $\rho_{\mathrm{A}}(r)$ and $\rho_{\mathrm{R}}(r)$. This begs the question: how can we bound $\rho_{\mathrm{A}}(r)$ and $\rho_{\mathrm{R}}(r)$? It turns out we can do so using the Gittins game where the initial state is X_{A}, the random state of a new arrival.

Lemma 8.3. *For all $r \geq 0$, we have $\rho_{\mathrm{A}}(r) + \rho_{\mathrm{R}}(r) \leq \lambda\, \mathbf{E}[\mathsf{game}(X_{\mathrm{A}}, r)]$.*

We defer the proof of Lemma 8.3 to Appendix B.1. Of course, the question of how to bound $\mathbf{E}[\mathsf{game}(X_{\mathrm{A}}, r)]$ remains. It turns out we will need both upper and lower bounds, which the following lemma provides. It is convenient to state the lower bound in terms of the *excess* S_{e} of the service requirement distribution S, which is the distribution with tail

$$\mathbf{P}[S_{\mathrm{e}} > r] = \frac{1}{\mathbf{E}[S]} \int_r^\infty \mathbf{P}[S > s]\, \mathrm{d}s.$$

Lemma 8.4. *For all $r \geq 0$, we have $\mathbf{E}[S]\, \mathbf{P}[S_{\mathrm{e}} \leq r] = \mathbf{E}[\mathsf{game}(X_{\mathrm{A}}, r)] \leq \min\{r, \mathbf{E}[S]\}$.*

Proof. The upper bound follows immediately from Lemma 5.2. Suppose we had perfect knowledge of an arriving job's service requirement S. Then we would achieve optimal cost $\min\{r, S\}$ in the Gittins game, serving the job to completion if $S < r$ and stopping immediately otherwise. This means $\mathbf{E}[\mathsf{game}(X_{\mathrm{A}}, r)] = \mathbf{E}[\min\{r, S\}] = \mathbf{E}[S]\, \mathbf{P}[S_e \leq r]$ in the perfect-information case. Having less information clearly increases the optimal cost, so $\mathbf{E}[\mathsf{game}(X_{\mathrm{A}}, r)] \geq \mathbf{E}[S]\, \mathbf{P}[S_e \leq r]$. □

8.3 Bounding Response Time

We now have all the tools in place to bound mean response time in the M/G/k.

Lemma 8.5. *For all $r \geq 0$,*

$$\int_0^\infty \frac{\mathbf{E}\big[(1 - B(r))\, W(r)\big] + \lambda_{\mathrm{R}}(r)\, \mathbf{E}_r\big[S(X_{\mathrm{R}}(r), r)\, W(r)\big]}{1 - \rho_{\mathrm{A}}(r)} \frac{1}{r^2}\, \mathrm{d}r$$
$$\leq (k-1)\left(\min\left\{\frac{\rho}{1-\rho}, 1\right\} + \int_0^\infty \min\left\{2\lambda, \frac{\rho - \rho_{\mathrm{A}}(r)}{1 - \rho_{\mathrm{A}}(r)} \frac{1}{r}\right\} \mathrm{d}r\right).$$

Proof. Applying Lemmas 8.1 and 8.2 yields

$$\int_0^\infty \frac{\mathbf{E}\big[(1 - B(r))\, W(r)\big] + \lambda_{\mathrm{R}}(r)\, \mathbf{E}_r\big[S(X_{\mathrm{R}}(r), r)\, W(r)\big]}{1 - \rho_{\mathrm{A}}(r)} \frac{1}{r^2}\, \mathrm{d}r$$
$$\leq (k-1) \int_0^\infty \frac{\min\{\rho, 1-\rho\}}{1 - \rho_{\mathrm{A}}(r)} \mathbf{E}[\mathsf{service}(X_1, r) \mid X_1 \neq \top \text{ and } X_2 = \top] \frac{1}{r^2}\, \mathrm{d}r$$
$$+ (k-1) \int_0^\infty \frac{\min\{\rho_{\mathrm{A}}(r) + \rho_{\mathrm{R}}(r), \rho - \rho_{\mathrm{A}}(r) - \rho_{\mathrm{R}}(r)\} + \rho_{\mathrm{R}}(r)}{1 - \rho_{\mathrm{A}}(r)} \frac{1}{r}\, \mathrm{d}r.$$

To bound the first term, we compute

$$\int_0^\infty \frac{\min\{\rho, 1-\rho\}}{1 - \rho_{\mathrm{A}}(r)} \mathbf{E}[\mathsf{service}(X_1, r) \mid X_1 \neq \top \text{ and } X_2 = \top] \frac{1}{r^2}\, \mathrm{d}r$$
$$\leq \int_0^\infty \min\left\{\frac{\rho}{1-\rho}, 1\right\} \mathbf{E}[\mathsf{service}(X_1, r) \mid X_1 \neq \top \text{ and } X_2 = \top] \frac{1}{r^2}\, \mathrm{d}r \quad \text{[by Lemmas 8.3 and 8.4]}$$
$$= \min\left\{\frac{\rho}{1-\rho}, 1\right\} \mathbf{E}\left[\int_0^\infty \frac{\mathsf{service}(X_1, r)}{r^2}\, \mathrm{d}r \,\middle|\, X_1 \neq \top \text{ and } X_2 = \top\right]$$
$$= \min\left\{\frac{\rho}{1-\rho}, 1\right\} \quad \text{[by Lemma 6.2]}.$$

To bound the second term, note that whenever the $\rho_A(r) + \rho_R(r)$ branch of the minimum is used, we have $\rho_A(r) + \rho_R(r) \le 1/2$, so we can bound

$$\begin{aligned}\frac{\rho_A(r) + 2\,\rho_R(r)}{1 - \rho_A(r)} &= \frac{2(\rho_A(r) + \rho_R(r)) - \rho_A(r)}{1 - \rho_A(r)} \\ &\le 2(\rho_A(r) + \rho_R(r)) \\ &\le 2\lambda r. \qquad \text{[by Lemmas 8.3 and 8.4]}\end{aligned}$$

□

The next lemma is a straightforward computation that further bounds the right-hand side of Lemma 8.5. We prove it in Appendix B.2.

Lemma 8.6. *For all $b \ge a \ge 0$,*

$$\begin{aligned}&\int_0^\infty \min\left\{2\lambda, \frac{\rho - \rho_A(r)}{1 - \rho_A(r)} \frac{1}{r}\right\} \mathrm{d}r \\ &\quad\le \frac{2a\rho}{\mathbf{E}[S]} + \rho \log \frac{b}{a} + \log\left(1 + \frac{\rho}{1-\rho}\,\mathbf{P}[S_e > b]\right) + \frac{\rho}{1-\rho}\int_b^\infty \frac{\mathbf{P}[S_e > r]}{r}\,\mathrm{d}r.\end{aligned}$$

Lemma 8.7. *For all $\alpha > 1$, if $\mathbf{E}[S^\alpha] < \infty$, then*

$$\int_0^\infty \min\left\{2\lambda, \frac{\rho - \rho_A(r)}{1 - \rho_A(r)} \frac{1}{r}\right\} \mathrm{d}r \le \frac{\rho}{\alpha - 1}\left(\log \frac{1}{1-\rho} + \log \frac{\mathbf{E}[S^\alpha]}{\alpha\,\mathbf{E}[S]^\alpha} + 1\right) + \left(1 + \frac{2}{e}\right)\rho + \log(1+\rho).$$

Proof. Our overall approach is to apply Lemma 8.6, choosing b such that $\mathbf{P}[S_e > b] \le 1 - \rho$. We begin by using Markov's inequality and the fact that $\mathbf{E}[S^\alpha]$ is finite to bound $\mathbf{P}[S_e > r]$:

$$\mathbf{P}[S_e > r] \le \frac{\mathbf{E}[S_e^{\alpha-1}]}{r^{\alpha-1}} = c_\alpha \left(\frac{\mathbf{E}[S]}{r}\right)^{\alpha-1}, \tag{8.1}$$

where $c_\alpha = \mathbf{E}[S_e^{\alpha-1}]/\mathbf{E}[S]^{\alpha-1} = \mathbf{E}[S^\alpha]/(\alpha\,\mathbf{E}[S]^\alpha)$. Therefore, if we choose

$$b = \mathbf{E}[S]\left(\frac{c_\alpha}{1-\rho}\right)^{1/(\alpha-1)}, \tag{8.2}$$

then $\mathbf{P}[S_e > b] \le 1 - \rho$.

It remains to choose $a \in [0, b]$. Because $c_\alpha \ge 1/\alpha$ and $\alpha > 1$, we have $b > \mathbf{E}[S]/e$, so we choose

$$a = \frac{\mathbf{E}[S]}{e}. \tag{8.3}$$

We conclude by plugging our choices of a and b into Lemma 8.6, bounding each term in turn.

- By (8.3), the first term is $2a\rho/\mathbf{E}[S] = 2\rho/e$.
- By (8.2) and (8.3), the second term is

$$\rho \log \frac{b}{a} = \rho + \rho \log \frac{b}{\mathbf{E}[S]} = \rho + \frac{\rho}{\alpha - 1}\left(\log \frac{1}{1-\rho} + \log c_\alpha\right).$$

- By (8.1) and (8.2), we have $\mathbf{P}[S_e > b] \le 1 - \rho$, so the third term is bounded by $\log(1+\rho)$.
- By (8.1) and (8.2), the fourth term is bounded by

$$\frac{\rho}{1-\rho}\int_b^\infty \frac{\mathbf{P}[S_e > r]}{r}\,\mathrm{d}r \le \frac{\rho}{1-\rho}\int_b^\infty \frac{c_\alpha\,\mathbf{E}[S]^{\alpha-1}}{r^\alpha}\,\mathrm{d}r = \frac{\rho}{1-\rho}\,\frac{c_\alpha\,\mathbf{E}[S]^{\alpha-1}}{\alpha-1}\,\frac{1}{b^{\alpha-1}} = \frac{\rho}{\alpha-1}.$$

□

8.4 Proofs of Main Theorems

Theorem 8.8. *For all $\alpha > 1$, if $\mathbf{E}[S^\alpha] < \infty$, then the mean response time gap between the M/G/k and M/G/1 under Gittins is at most*

$$\mathbf{E}[T_k] - \mathbf{E}[T_1] \leq (k-1)\,\mathbf{E}[S]\left(\frac{1}{\alpha-1}\left(\log\frac{1}{1-\rho} + \log\frac{\mathbf{E}[S^\alpha]}{\alpha\,\mathbf{E}[S]^\alpha} + 1\right) + \min\left\{\frac{1}{1-\rho}, \frac{1}{\rho}\right\} + 1 + \frac{2}{e} + \frac{\log(1+\rho)}{\rho}\right).$$

Proof. The proof combines the following pieces:

- Theorem 6.3 turns the mean response time gap into an integral of mean relevant work gaps.
- Theorem 7.2 gives a formula for the mean relevant work gap.
- Lemmas 8.5 and 8.7 bound the resulting integral.

Specifically, recalling $\rho = \lambda\,\mathbf{E}[S]$, we compute

$$\begin{aligned}
&\mathbf{E}[T_k] - \mathbf{E}[T_1] \\
&= \frac{\mathbf{E}[S]}{\rho}\int_0^\infty \frac{\mathbf{E}[W_k(r)] - \mathbf{E}[W_1(r)]}{r^2}\,\mathrm{d}r && \text{[by Theorem 6.3]} \\
&= \frac{\mathbf{E}[S]}{\rho}\int_0^\infty \frac{\mathbf{E}\left[(1-B(r))\,W(r)\right] + \lambda_{\mathrm{R}}(r)\,\mathbf{E}_r\left[S(X_{\mathrm{R}}(r), r)\,W(r)\right]}{1-\rho_{\mathrm{A}}(r)}\frac{1}{r^2}\,\mathrm{d}r && \text{[by Theorem 7.2]} \\
&\leq (k-1)\,\mathbf{E}[S]\left(\min\left\{\frac{1}{1-\rho}, \frac{1}{\rho}\right\} + \frac{1}{\rho}\int_0^\infty \min\left\{2\lambda, \frac{\rho-\rho_{\mathrm{A}}(r)}{1-\rho_{\mathrm{A}}(r)}\frac{1}{r}\right\}\mathrm{d}r\right) && \text{[by Lemma 8.5]} \\
&\leq (k-1)\,\mathbf{E}[S]\left(\frac{1}{\alpha-1}\left(\log\frac{1}{1-\rho} + \log\frac{\mathbf{E}[S^\alpha]}{\alpha\,\mathbf{E}[S]^\alpha} + 1\right)\right. \\
&\qquad\qquad \left. + \min\left\{\frac{1}{1-\rho}, \frac{1}{\rho}\right\} + 1 + \frac{2}{e} + \frac{\log(1+\rho)}{\rho}\right). && \text{[by Lemma 8.7]}
\end{aligned}$$

□

Theorem 1.1. *For all $\alpha > 1$, if $\mathbf{E}[S^\alpha] < \infty$, then the mean response time gap between the M/G/k and M/G/1 under Gittins is at most*

$$\mathbf{E}[T_k] - \mathbf{E}[T_1] \leq (k-1)\,\mathbf{E}[S]\left(\frac{1}{\alpha-1}\left(\log\frac{1}{1-\rho} + \log\frac{\mathbf{E}[S^\alpha]}{\alpha\,\mathbf{E}[S]^\alpha} + 1\right) + 4.547\right).$$

Proof. The bound in Theorem 8.8 is maximized at $\rho = 1/2$. □

In the special case of $\alpha = 2$, we can simplify and slightly improve upon the bound in Theorem 1.1.

Theorem 1.2. *If $C_S^2 = \mathbf{Var}[S^2]/\mathbf{E}[S]^2 < \infty$, then the mean response time gap between the M/G/k and M/G/1 under Gittins is at most*

$$\mathbf{E}[T_k] - \mathbf{E}[T_1] \leq (k-1)\,\mathbf{E}[S]\left(\log\frac{1}{1-\rho} + \log(1+C_S^2) + 4.811\right).$$

Proof. We make one small change to the proof of Theorem 8.8, resulting in a slightly smaller bound. When $\alpha \geq 2$, one can choose $a = \mathbf{E}[S]/2$ instead of $a = \mathbf{E}[S]/e$ in the proof of Lemma 8.7. Doing so replaces the $1 + 2/e$ factor with $1 + \ln(2)$. □

Our last main result, Theorem 1.3, shows that Gittins minimizes mean response time in the heavy-traffic limit. It suffices to show that the mean response time gap in Theorem 1.1 is dominated by the M/G/1 mean response time under Gittins. The proof of Theorem 1.3, deferred to Appendix B.2, thus amounts to showing that the condition it imposes on S implies $\mathbf{E}[T_1] = \omega(\log(1/(1-\rho)))$.

9 CONCLUSION

We study the Gittins policy in the M/G/k. The Gittins policy is known to have optimal mean response time for the single-server $k = 1$ case, but the problem was previously open for the much more difficult $k \geq 2$ case. We prove that the Gittins policy has near-optimal mean response time in the M/G/k at all loads (Theorem 1.1) and is optimal in the heavy traffic limit (Theorem 1.3). Our results hold under an extremely broad job model and under extremely general conditions on the service requirement distribution.

Our proof bounds the additive gap in mean response time between the Gittins policy in the M/G/k and in an M/G/1 where the server runs k times faster than the M/G/k servers. Our proof works by directly relating response time in the M/G/k to relevant work, without going through the intervening step of the tagged-job method. We combine this with a new relevant work decomposition law, which we prove using Palm calculus, which bounds the relevant work gap between the M/G/k and M/G/1. This novel technique may be applicable far beyond the M/G/k setting, and in particular Theorem 6.3 is not specific to the M/G/k.

We show that the additive gap between the mean response time of Gittins in the M/G/k and in the M/G/1 scales as $O(k \log(1/(1-\rho)))$. A direction for future work would be to show that this bound is tight, or to further tighten the bound. In addition, even if our bound on the Gittins policy was proven to be tight, it would be open whether any policy can achieve a gap smaller than $O(k \log(1/(1-\rho)))$.

ACKNOWLEDGMENTS

We thank Justin Whitehouse for helpful discussions that improved the quality of Appendix D. We also thank the anonymous referees for their comments and suggestions.

This work was supported by NSF grants CMMI-1938909, XPS-1629444, and CSR-1763701; and a Google 2020 Faculty Research Award.

REFERENCES

[1] 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.

[2] 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.

[3] Dimitris Bertsimas and José Niño-Mora. 1999. Optimization of Multiclass Queueing Networks with Changeover Times Via the Achievable Region Approach: Part II, The Multi-Station Case. *Mathematics of Operations Research* 24, 2 (1999), 331–361.

[4] Onno J Boxma. 1989. Workloads and waiting times in single-server systems with multiple customer classes. *Queueing Systems* 5, 1-3 (1989), 185–214.

[5] 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.

[6] Ioana Dumitriu, Prasad Tetali, and Peter Winkler. 2003. On playing golf with two balls. *SIAM Journal on Discrete Mathematics* 16, 4 (2003), 604–615.

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

[8] Kevin D Glazebrook. 2003. An analysis of Klimov's problem with parallel servers. *Mathematical Methods of Operations Research* 58, 1 (2003), 1–28.

[9] Kevin D Glazebrook and José Niño-Mora. 2001. Parallel scheduling of multiclass M/M/m queues: Approximate and heavy-traffic optimization of achievable performance. *Operations Research* 49, 4 (2001), 609–623.

[10] Isaac Grosof, Ziv Scully, and Mor Harchol-Balter. 2018. SRPT for Multiserver Systems. *Performance Evaluation* 127–128 (2018), 154–175.

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

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

[13] Minghong Lin, Adam Wierman, and Bert Zwart. 2010. The average response time in a heavy-traffic SRPT queue. In *ACM SIGMETRICS Performance Evaluation Review*, Vol. 38. ACM, 12–14.
[14] Paul Milgrom and Ilya Segal. 2002. Envelope Theorems for Arbitrary Choice Sets. *Econometrica* 70, 2 (2002), 583–601.
[15] Michael Mitzenmacher. 2020. Scheduling with Predictions and the Price of Misprediction. In *11th Innovations in Theoretical Computer Science Conference (ITCS 2020) (Leibniz International Proceedings in Informatics (LIPIcs))*, Thomas Vidick (Ed.), Vol. 151. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany, 14:1–14:18.
[16] Masakiyo Miyazawa. 1994. Rate conservation laws: a survey. *Queueing Systems* 15, 1-4 (1994), 1–58.
[17] Goran Peskir and Albert Shiryaev. 2006. *Optimal Stopping and Free-Boundary Problems*. Springer.
[18] Ziv Scully, Isaac Grosof, and Mor Harchol-Balter. 2020. Optimal Multiserver Scheduling with Unknown Job Sizes in Heavy Traffic. *Performance Evaluation* (2020). To appear.
[19] Ziv Scully, Mor Harchol-Balter, and Alan Scheller-Wolf. 2018. Optimal Scheduling and Exact Response Time Analysis for Multistage Jobs. (2018). arXiv:1805.06865
[20] 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.
[21] Albert Shiryaev. 2007. *Optimal Stopping Rules*. Vol. 8. Springer.
[22] Richard Weber. 1992. On the Gittins index for multiarmed bandits. *The Annals of Applied Probability* 2, 4 (1992), 1024–1033.
[23] Peter Whittle. 1980. Multi-armed bandits and the Gittins index. *Journal of the Royal Statistical Society. Series B (Methodological)* (1980), 143–149.
[24] Ronald W. Wolff. 1982. Poisson arrivals see time averages. *Operations Research* 30, 2 (1982), 223–231.

A SUMMARY OF NOTATION

Below we summarize the main notation used in the paper. We sometimes add a subscript k, as in $\mathbf{E}[T_k]$, to emphasize dependence on the number of servers, such as when discussing the mean response time gap $\mathbf{E}[T_k] - \mathbf{E}[T_1]$ between the M/G/k and M/G/1.

M/G/k notation:

$$\begin{aligned} k &= \text{number of servers, each speed } 1/k, \\ \lambda &= \text{arrival rate}, \\ S &= \text{service requirement}, \\ S_{\mathrm{e}} &= \text{excess of service requirement}, \\ \rho &= \text{load} = \lambda\,\mathbf{E}[S], \\ N &= \text{number of jobs in the system}, \\ T &= \text{response time}. \end{aligned}$$

Job state notation:

$$\begin{aligned} \mathbb{X} &= \text{job state space}, \\ X_i &= \text{state of } i\text{th job in system, where the first } k \text{ jobs are in service}, \\ X_{\mathrm{A}} &= \text{random state of a new arrival}, \\ \top &= \text{departure state, representing the absence of a job}. \end{aligned}$$

Gittins notation:

$$\begin{aligned} \mathsf{rank}(x) &= \text{Gittins rank of state } x, \\ S(x,r) &= \text{service needed for job in state } x \text{ to complete or reach rank } r, \\ \mathsf{service}(x,r) &= \mathbf{E}[S(x,r)], \\ \mathsf{undone}(x,r) &= \mathbf{P}[\text{job in state } x \text{ completes before reaching rank } r], \\ \mathsf{game}(x,r) &= \mathsf{service}(x,r) + r\,\mathsf{undone}(x,r). \end{aligned}$$

Relevant work notation:

$$W(r) = r\text{-relevant work in system} = \sum_{i=1}^{\infty} S(X_i, r),$$
$$B(r) = \text{fraction of servers busy with } r\text{-relevant jobs} = \frac{1}{k}\sum_{i=1}^{k} \mathbb{1}(\mathsf{rank}(X_i) < r).$$

Fresh and recycled jobs notation:

$$\rho_{\mathrm{A}}(r) = \text{relevant load due to fresh jobs} = \lambda\, \mathbf{E}[S(X_{\mathrm{A}}, r)],$$
$$\lambda_{\mathrm{R}}(r) = \text{average rate of } r\text{-recyclings},$$
$$\mathbf{E}_r[\cdot] = \text{expectation sampled just before } r\text{-recyclings},$$
$$X_{\mathrm{R}}(r) = \text{random state of a job just after its } r\text{-recycling},$$
$$\rho_{\mathrm{R}}(r) = \text{relevant load due to recycled jobs} = \lambda_{\mathrm{R}}(r)\, \mathbf{E}_r[S(X_{\mathrm{R}}(r), r)].$$

B DEFERRED PROOFS

B.1 Gittins Game Lemmas

Lemma 5.3. *For all $x \in \mathbb{X}$, the function $r \mapsto \mathsf{game}(x, r)$ is continuous, concave, and differentiable almost everywhere with*

$$\frac{\mathrm{d}}{\mathrm{d}r}\,\mathsf{game}(x, r) = \mathsf{undone}(x, r).$$

Proof. By (5.1), we know $\mathsf{game}(x, r, \mathbb{Y})$ is linear in r. Because a minimum of linear functions is concave, $\mathsf{game}(x, r) = \min_{\mathbb{Y}} \mathsf{game}(x, r, \mathbb{Y})$ is concave in r, so it is differentiable almost everywhere. Concavity implies continuity for $r > 0$, and Lemma 5.2 implies continuity at $r = 0$. An envelope theorem of Milgrom and Segal [14, Theorem 1] implies that where the derivative exists,

$$\frac{\mathrm{d}}{\mathrm{d}r}\,\mathsf{game}(x, r) = \frac{\mathrm{d}}{\mathrm{d}r}\,\mathsf{game}(x, r, \mathbb{Y}^*(s))\bigg|_{s=r},$$

and by (5.2),

$$\frac{\mathrm{d}}{\mathrm{d}r}\,\mathsf{game}(x, r, \mathbb{Y}^*(s))\bigg|_{s=r} = \mathsf{undone}(x, \mathbb{Y}^*(r)) = \mathsf{undone}(x, r). \qquad \square$$

Lemma 8.3. *For all $r \geq 0$, we have $\rho_{\mathrm{A}}(r) + \rho_{\mathrm{R}}(r) \leq \lambda\, \mathbf{E}[\mathsf{game}(X_{\mathrm{A}}, r)]$.*

Proof. By Definition 7.1, we can think of $\rho_{\mathrm{A}}(r) + \rho_{\mathrm{R}}(r)$ as the load due to r-fresh and r-recycled jobs. Our approach is to bound the expected amount of service a job receives while it is r-fresh or r-recycled. Specifically, it suffices to show that while serving a job to completion starting at any initial state x, the expected amount of service during which the job is r-relevant is at most $\mathsf{game}(x, r)$.

The key idea is to consider the following "bank account" variant of the Gittins game. We start the game with a bank account storing value u. Whenever the job is *irrelevant*, we may spend the bank account value at rate 1 to serve the job without incurring the usual cost. We cannot spend from the bank account while the job is relevant. If we stop serving the job before it completes, we still pay the penalty r, and any leftover value in the bank account is worthless.

Let $\mathsf{game}_u(x, r)$ be the optimal cost in the bank account Gittins game starting with value u in the bank account. We have $\mathsf{game}_0(x, r) = \mathsf{game}(x, r)$, and clearly $\mathsf{game}_u(x, r)$ is nonincreasing in u, because we can always choose not to spend from the bank account. It thus suffices to show that

$$\mathsf{game}_\infty(x, r) \coloneqq \lim_{u \to \infty} \mathsf{game}_u(x, r)$$

is the expected amount of service during which the job is r-relevant as it is served from initial state x to completion. This amounts to showing that when $u = \infty$, it is always optimal to serve the job whenever it is r-relevant. Lemma 5.4 implies that when $u = 0$, it is optimal to serve the job while it is r-relevant. Increasing u only makes serving the job more appealing, so serving the job is optimal for any value of u.

Having analyzed the bank account Gittins game, we are finally ready to bound $\rho_A(r) + \rho_R(r)$:

$$\rho_A(r) + \rho_R(r) = \lambda\, \mathbf{E}[\mathsf{game}_\infty(X_A, r)] \le \lambda\, \mathbf{E}[\mathsf{game}_0(X_A, r)] = \mathsf{game}_A(r). \qquad \square$$

There is another very elegant proof of Lemma 8.3 based on a "prevailing reward" style of argument that is common in the Gittins literature [6, 22]. One usually thinks of a job's prevailing reward as its maximum current or past Gittins rank. To prove Lemma 8.3, one can use a variant of the prevailing reward that looks back only to a job's most recent recycling instead of its entire past.

B.2 Computations for Proofs of Main Theorems

We define

$$\begin{aligned}
\mathsf{service}_A(r) &\coloneqq \mathbf{E}[\mathsf{service}(X_A, r)] = \mathbf{E}[S(X_A, r)],\\
\mathsf{undone}_A(r) &\coloneqq \mathbf{E}[\mathsf{undone}(X_A, r)],\\
\mathsf{game}_A(r) &\coloneqq \mathbf{E}[\mathsf{game}(X_A, r)] = \mathsf{service}_A(r) + r\,\mathsf{undone}_A(r).
\end{aligned}$$

Lemma B.1. *For all $r \ge 0$,*

$$\frac{\rho - \rho_A(r)}{1 - \rho_A(r)} \frac{1}{r} \le \frac{\mathrm{d}}{\mathrm{d}r} \log\left(\frac{1}{1 - \lambda\, \mathsf{game}_A(r)}\right) + \frac{\rho}{1-\rho} \frac{\mathbf{P}[S_e > r]}{r}.$$

Proof. We compute

$$\begin{aligned}
\frac{\rho - \rho_A(r)}{1 - \rho_A(r)} \frac{1}{r} &= \frac{\rho - \lambda(\mathsf{game}_A(r) - r\,\mathsf{undone}_A(r))}{1 - \lambda(\mathsf{game}_A(r) - r\,\mathsf{undone}_A(r))} \frac{1}{r}\\
&\le \frac{\lambda\,\mathsf{undone}_A(r)}{1 - \lambda\,\mathsf{game}_A(r)} + \frac{\rho - \lambda\,\mathsf{game}_A(r)}{1 - \lambda\,\mathsf{game}_A(r)} \frac{1}{r}\\
&\le \frac{\lambda \frac{\mathrm{d}}{\mathrm{d}r}\mathsf{game}_A(r)}{1 - \lambda\,\mathsf{game}_A(r)} + \frac{\rho - \rho\,\mathbf{P}[S_e \le r]}{1-\rho} \frac{1}{r} \qquad \text{[by Lemmas 5.3 and 8.4]}\\
&= \frac{\mathrm{d}}{\mathrm{d}r} \log\left(\frac{1}{1 - \lambda\,\mathsf{game}_A(r)}\right) + \frac{\rho}{1-\rho} \frac{\mathbf{P}[S_e > r]}{r}. \qquad \square
\end{aligned}$$

Lemma 8.6. *For all $b \ge a \ge 0$,*

$$\begin{aligned}
&\int_0^\infty \min\left\{2\lambda, \frac{\rho - \rho_A(r)}{1 - \rho_A(r)} \frac{1}{r}\right\} \mathrm{d}r\\
&\quad\le \frac{2a\rho}{\mathbf{E}[S]} + \rho \log\frac{b}{a} + \log\left(1 + \frac{\rho}{1-\rho}\,\mathbf{P}[S_e > b]\right) + \frac{\rho}{1-\rho} \int_b^\infty \frac{\mathbf{P}[S_e > r]}{r}\,\mathrm{d}r.
\end{aligned}$$

Proof. We split the integration region into three intervals: $(0, a)$, (a, b), and (b, ∞).

- For $(0, a)$, we bound the integrand by 2λ.
- For (a, b), we bound the integrand by ρ/r.
- For (b, ∞), we bound the integrand using Lemma B.1.

Finally, we bound one of the terms from the (b, ∞) region that comes from Lemma B.1 as follows:

$$\begin{aligned}\int_b^\infty \frac{\mathrm{d}}{\mathrm{d}r} \log\left(\frac{1}{1-\lambda\,\mathrm{game}_\mathrm{A}(r)}\right) \mathrm{d}r &= \log \frac{1-\lambda\,\mathrm{game}_\mathrm{A}(b)}{1-\rho} \\ &\le \log \frac{1-\rho\,\mathbf{P}[S_\mathrm{e} \le b]}{1-\rho} && \text{[by Lemma 8.4]} \\ &= \log\left(1 + \frac{\rho}{1-\rho}\,\mathbf{P}[S_\mathrm{e} > b]\right). \end{aligned}$$

□

Theorem 1.3. *If* $\mathbf{E}[S^2(\log S)^+] < \infty$*, then under Gittins,*

$$\lim_{\rho\to 1} \frac{\mathbf{E}[T_k]}{\mathbf{E}[T_1]} = 1,$$

so Gittins minimizes mean response time in the M/G/k in the heavy-traffic limit.

Proof. Because Gittins minimizes mean response time in the M/G/1 (Theorem 7.3), it suffices to show that $\mathbf{E}[T_1] = \omega(\log(1/(1-\rho)))$ in the $\rho \to 1$ limit.[10] Lin et al. [13] analyze the mean response time of SRPT in the M/G/1, which is a lower bound on that of Gittins. A result of theirs implies [13, Lemma 5]

$$\mathbf{E}[T_1] = \Omega\left(\frac{1}{(1-\rho)\,G^{-1}(\rho)}\right),$$

where $G(r) = \mathbf{E}[S\,\mathbb{1}(S \le r)]/\mathbf{E}[S]$. We therefore want to show

$$\frac{1}{G^{-1}(\rho)} = \omega\left((1-\rho)\log\frac{1}{1-\rho}\right),$$

which amounts to showing that in the $r \to \infty$ limit,

$$(1-G(r))\log\frac{1}{1-G(r)} = o\left(\frac{1}{r}\right).$$

It actually suffices to show $1 - G(r) = o(1/(r\log r))$, as then

$$(1-G(r))\log\frac{1}{1-G(r)} = o\left(\frac{\log r + \log\log r}{r\log r}\right) = o\left(\frac{1}{r}\right).$$

We now use the $\mathbf{E}[S^2(\log S)^+] < \infty$ assumption to prove $1 - G(r) = o(1/(r\log r))$. The first step is to express $1 - G(r)$ in terms of S and S_e:

$$1 - G(r) = \frac{\mathbf{E}[S\,\mathbb{1}(S > r)]}{\mathbf{E}[S]} = \frac{\mathbf{E}[(S-r)^+]}{\mathbf{E}[S]} = \frac{r}{\mathbf{E}[S]}\,\mathbf{P}[S > r] + \mathbf{P}[S_\mathrm{e} > r].$$

We show that both terms are $o(1/(r\log r))$. For the first term, finiteness of $\mathbf{E}[S^2(\log S)^+]$ implies

$$r^2(\log r)^+\,\mathbf{P}[S > r] \le \mathbf{E}[S^2(\log S)^+\,\mathbb{1}(S > r)] = \mathbf{E}[S^2(\log S)^+] - \mathbf{E}[S^2(\log S)^+\,\mathbb{1}(S \le r)] = o(1).$$

We can bound the second term similarly because

$$\mathbf{E}[S]\,\mathbf{E}[S_\mathrm{e}(\log S_e)^+] = \mathbf{E}\left[\mathbb{1}(S > 1)\int_1^S s\log s\,\mathrm{d}s\right] = \mathbf{E}\left[\mathbb{1}(S > 1)\left(\frac{1}{2}S^2\log S - \frac{1}{4}S^2 + \frac{1}{4}\right)\right] < \infty.$$

□

[10] Formally, by the $\rho \to 1$ limit, we mean the $\lambda \to 1/\mathbf{E}[S]$ limit while all other parameters of the system model are constant.

C COMPUTING THE GITTINS RANK

In general, computing the Gittins rank of a state may be very difficult. This is because the Gittins rank formula in Definition 4.1 requires optimizing over subsets of states. With that said, in some special cases, the rank computation can be simplified. The simplest case is that of finite state spaces, for which the Gittins rank function can be computed in polynomial time [5].

The examples in Section 3.3 also admit relatively simple rank computations. In each case, the optimization over stopping sets in Definition 4.1 reduces to a finite-dimensional optimization problem or simpler. We discuss each example in turn.

- *Perfect information* (Example 3.1): Here a job's state x is its remaining size, and the optimizing stopping set is simply $\mathbb{Y} = \{\top\}$.
- *Zero information* (Example 3.2): Here the rank computation reduces to a single-dimensional optimization problem, namely that shown in (4.1).
- *Size estimates* (Example 3.3): Much like the previous case, there the rank computation reduces to a single-dimensional optimization problem. The optimization for state (m, s), where m is the job's estimated size and s is its attained service, is very similar to that in (4.1), except we condition the expectations on $M = m$.
- *Multistage jobs* (Example 3.4): Here the rank computation reduces to a finite-dimensional optimization problem. Specifically, when in state (i, s_i), meaning we have are in stage i and have received s_i units of service in this stage, choosing a stopping set amounts to choosing the following values:
  - a cutoff $t > s_i$ such that we stop if we reach state (i, t); and
  - a cutoff $t_j \geq 0$ for each stage j reachable from i such that we stop if we reach state (j, t_j).

D TECHNICAL FOUNDATIONS OF THE JOB MODEL AND THE GITTINS GAME

We have thus far not discussed any technical conditions that the job Markov process must satisfy. The details of these conditions do not affect the bulk of our proof in Sections 6–8, which uses definitions and properties from Sections 4 and 5 in a largely "black-box" manner. The purpose of this appendix is to sketch the technical conditions we implicitly assume on the job model over the course of Sections 4 and 5. We emphasize that these assumptions are satisfied for virtually all cases of practical interest, including the examples in Section 3.3.

We begin by giving a more formal presentation of the Gittins game and Gittins rank (Appendix D.1). This presentation works for any job Markov process, but it complicates the definition of relevant work, so we give sufficient conditions on the job Markov process such that Definition 6.1 gives the right definition of relevant work (Appendix D.2).

D.1 Defining the Gittins Game as an Optimal Stopping Problem

In Section 5 we define $\mathsf{game}(x, r, \mathbb{Y}) \coloneqq \mathsf{service}(x, \mathbb{Y}) + \mathsf{undone}(x, \mathbb{Y})$ for stopping sets $\mathbb{Y} \subseteq \mathbb{X}$, and we define the Gittins game as the optimization problem $\mathsf{game}(x, r) \coloneqq \inf_{\mathbb{Y} \subseteq \mathbb{X}} \mathsf{game}(x, r, \mathbb{Y})$. However, these definitions ignore two technicalities. First, it may be that $\mathsf{service}(x, \mathbb{Y})$ and $\mathsf{undone}(x, \mathbb{Y})$ are not well defined for some sets $\mathbb{Y}$. Second, we neglect potential strategies that are not of the form "run the job until the state enters a stopping set".

To fix both of these issues, we redefine the Gittins game as an optimization over *stopping times*. Letting τ range over stopping times that are almost surely finite, define

$$\begin{aligned} \mathsf{service}(x, \tau) &\coloneqq \mathbf{E}[\tau \mid X(0) = x], \\ \mathsf{done}(x, \tau) &\coloneqq \mathbf{P}[X(\tau) = \top \mid X(0) = x], \\ \mathsf{undone}(x, \tau) &\coloneqq 1 - \mathsf{done}(x, \tau). \end{aligned}$$

We can now frame the Gittins game as an optimization over stopping times:

$$\begin{aligned} \mathsf{game}(x, r, \tau) &\coloneqq \mathsf{service}(x, \tau) + \mathsf{undone}(x, \tau), \\ \mathsf{game}(x, r) &\coloneqq \inf_{\tau} \mathsf{game}(x, r, \tau). \end{aligned} \tag{D.1}$$

We can also define the Gittins rank using stopping times:

$$\mathsf{rank}(x) \coloneqq \inf_{\tau > 0} \frac{\mathsf{service}(x, \tau)}{\mathsf{done}(x, \tau)}.$$

The definition of $\mathbb{Y}^*(r)$ remains the same, and the proof of Lemma 5.4 is easily adapted to use stopping times, so

$$\mathbb{Y}^*(r) \coloneqq \{x \in \mathbb{X} \mid \mathsf{game}(x, r) = r\} = \{x \in \mathbb{X} \mid \mathsf{rank}(x) \geq r\}.$$

Finally, define $\tau^*(r)$ to be the hitting time of $\mathbb{Y}^*(r)$:

$$\tau^*(r) = \inf\{s \geq 0 \mid X(s) \in \mathbb{Y}^*(r)\}.$$

D.2 Sufficient Conditions on the Job Model

The above definition of the Gittins game works without further assumptions on the job model. However, there remain technical issues related to relevant work (Definition 6.1). Throughout our proofs we make use of (6.1), which relates relevant work to the Gittins game. Having changed the definition of the Gittins game, we need to do one of two things: change the definition of relevant work, or give conditions such that our definition in Definition 6.1 suffices and (6.1) holds. We choose the latter approach.

Consider $S(x, r)$, the r-relevant remaining service requirement of a job in state x, which we can write as

$$S(x, r) = \left[\tau^*(r) \,\middle|\, X(0) = x\right].$$

We use $S(x, r)$ repeatedly throughout our proofs. We need it to be well defined, and we need to relate it to the solution to the Gittins game using (6.1), which states $\mathbf{E}[S(x, r)] = \mathsf{service}(x, \tau^*(r))$. The following condition ensures both of these properties:

> $\tau^*(r)$ is a stopping time and is optimal for the Gittins game, meaning it attains the infimum in (D.1).

Fortunately, the worry that the above condition might not hold is purely a technical concern. The intuition is that if we enter a state x with $\mathsf{game}(x, r) = r$, then paying the penalty r to stop achieves the optimal cost. It is thus unsurprising that one can show $\tau^*(r)$ is optimal under one of a variety of technical assumptions on the job Markov process [17, 21]. One such set of assumptions is the following:

- We assume the state space $\mathbb{X}$ is a locally compact Hausdorff space with a countable base endowed with the Borel σ-algebra. We further assume that the terminal state $\top$ is an isolated point.
  - This is not a significant restriction, as in cases where $\top$ is not isolated, such as Example 3.1, one can modify the state space to isolate $\top$, adding appropriate jumps to the trajectories.
- Let $\{\mathcal{F}(s)\}_{s \geq 0}$ be the natural filtration of $\{X(s)\}_{s \geq 0}$, meaning $\mathcal{F}(s)$ is the σ-algebra generated by $\{X(u) \mid u \in [0, s]\}$. We assume $\{\mathcal{F}(s)\}_{s \geq 0}$ is right-continuous, meaning $\mathcal{F}_s = \bigcap_{n=1}^{\infty} \mathcal{F}_{s+1/n}$.
  - Intuitively, this prevents scenarios in which we would learn information about a job by serving it for an infinitesimal amount of time. This is not a significant restriction, because one can augment the state space to include such information in the job's current state.
- We assume $\{X(s)\}_{s \geq 0}$ is a Feller process.

– This implies that $\{X(s)\}_{s\geq 0}$ obeys the strong Markov property and can be chosen to be right continuous with left limits.

The above conditions suffice because they allow optimal stopping problems for a wide class of continuous gain functions to be solved by the hitting time of a stopping set. See Shiryaev [21, Chapter 3, note especially Theorem 3] and Peskir and Shiryaev [17, Section 2, note especially (2.2.80)] for details. Specifically, one can frame the Gittins game as an optimal stopping problem for the process $\{(s, X(s))\}_{s\geq 0}$, where we augment a job's state with its attained service, with gain $g_r(s,x) = -(s + r\,\mathbb{1}(x \neq \top))$, which is continuous because $\top$ is isolated.

E WHAT IF THE RATE OF RECYCLINGS IS INFINITE?

We assume that $\lambda_{\mathrm{R}}(r) < \infty$ for ease of presentation (Section 7.2.1), and in particular for ease of understanding $\mathbf{E}_r[\cdot]$. However, there are instances of our model where $\lambda_{\mathrm{R}}(r) = \infty$, such as when the job Markov process involves Brownian motion. We can generalize our proof to these cases using more advanced concepts from Palm calculus, which we briefly outline below.

One can view $\mathbf{E}_r[\cdot]$ as the Palm expectation [16, Definition 2.2] with respect to the random measure that counts the cumulative number of r-recyclings. Unfortunately, $\mathbf{E}_r[\cdot]$ stops being useful when $\lambda_{\mathrm{R}}(r) = \infty$. For example, $\mathbf{E}_r[S(X_{\mathrm{R}}(r), r)] = 0$, which makes $\rho_{\mathrm{R}}(r) = \lambda_{\mathrm{R}}(r)\,\mathbf{E}_r[S(X_{\mathrm{R}}(r), r)]$ not well defined.

To fix this problem, instead of using $\mathbf{E}_r[\cdot]$, we use the Palm expectation with respect to the random measure $V(r)$ that measures the cumulative r-relevant work added by r-recyclings, denoted $\mathbf{E}_{V(r)}[\cdot]$. This entails making the following changes to our proofs:

- We define $\rho_{\mathrm{R}}(r)$ as the intensity of $V(r)$. That is, $\rho_{\mathrm{R}}(r)$ is still the average rate at which relevant work increases due to r-recyclings.
- We use $\mathbf{E}_{V(r)}[\cdot]$ instead of $\mathbf{E}_r[\cdot]$ in the statement and proof of Theorem 7.2. Specifically, the expected rate at which $W(r)^2$ increases due to recyclings is

$$\mathbf{E}_{V(r)}\left[\frac{(W(r) + \Delta V(r))^2 - W(r)^2}{\Delta V(r)}\right] = \mathbf{E}_{V(r)}[\Delta V(r)] + \mathbf{E}_{V(r)}[W(r)],$$

  where $\Delta V(r) = S(X_{\mathrm{R}}(r), r)$ is the amount of relevant work added by the recycling. As a result, we change the statement of Theorem 7.2 to

$$\mathbf{E}[W_k(r)] - \mathbf{E}[W_1(r)] = \frac{\mathbf{E}\big[(1 - B_k(r))\,W_k(r)\big] + \rho_{\mathrm{R}}(r)\,\mathbf{E}_{V(r)}[W_k(r)]}{1 - \rho_{\mathrm{A}}(r)}.$$

  That is, we have replaced $\lambda_{\mathrm{R}}(r)\,\mathbf{E}_r\big[S(X_{\mathrm{R}}(r), r)\,W_k(r)\big]$ with $\rho_{\mathrm{R}}(r)\,\mathbf{E}_{V(r)}[W_k(r)]$.
- The statement of Lemma 8.2 becomes

$$\mathbf{E}_{V(r)}[W_k(r)] \leq (k-1)r.$$

  The proof actually becomes simpler: we simply observe that at any moment when $V(r)$ increases, we must have at most $k-1$ relevant jobs, each of which contributes at most r relevant work by Lemma 5.2.

Received August 2020; revised September 2020; accepted October 2020