Transform Analysis of Preemption Overhead in the M/G/1

SHEFALI RAMAKRISHNA

1 INTRODUCTION

Preemptive scheduling policies, which allow pausing jobs mid-service, are widely used to let important jobs bypass less important ones that would otherwise delay their completion. A canonical example is Shortest Remaining Processing Time (SRPT), which always serves the job with the least remaining work [18]. Many works analyze *response time* (the time from a job's arrival to completion) in M/G/1 queues under such policies [14, 19, 20], shedding light on questions such as how preemption affects the mean and tail of response time, and whether preemption is unfair towards low-priority jobs.

In practice, preempting jobs incurs a delay called *preemption overhead* (e.g., due to context switches or cache reloads). Yet, with few exceptions (Section 2), queueing analyses typically ignore these costs, leaving it unclear how overhead affects preemption tradeoffs.

1.1 Contributions and techniques

We give the first transform-based response time analysis of the M/G/1 queue under preemptive priority scheduling with stochastic preemption overhead. Our model allows overhead to occur upon pause, resume, or both, with arbitrary distributions. For each priority class, we derive a recursive formula for the Laplace-Stieltjes transform of response time, from which all moments can be computed in closed form.

Our main analytical tool is the *job joint transform*, a multivariable transform capturing the joint distribution of a job's service time and the arrivals that occur during it. One of our key insights is that under preemptive priority scheduling, preemption overhead can be viewed as a type of *arrival-service dependency*. That is, when a high-priority job J_1 interrupts a lower-priority job J_2, the resulting overhead effectively extends J_2's service time.

One major obstacle is analyzing *busy periods* in systems with arrival-service dependency, which we overcome via the job joint transform. Recall that a busy period started by a set of jobs is the total service time of the initial set of "level 0" jobs, "level 1" arrivals that occur during service of level 0 jobs, "level 2" arrivals during level 1 jobs, etc. Busy period transforms play a critical role in many prior M/G/1 response time transform analyses as well as in the study of stability conditions. However, arrival-service dependency complicates busy period analysis. The job joint transform is a *multivariable* transform that packages this complex dependency into a single transform for each job. We then derive the busy period transforms as well as the specific job joint transforms for our preemption overhead model.

In this paper, we:

- Introduce the job joint transform for systems with arrival-service dependency (Section 3).
- Derive stability conditions and analyze busy periods for systems with arrival-service dependency (Sections 3.2 and 3.3).
- Express a flexible model of preemptive priority with preemption overhead as a system with arrival-service dependence, deriving the job joint transform for our model (Section 4).
- Derive our response time results for a system with preemption overhead (Section 5).

We view our work as a first step towards incorporating preemption overhead into SOAP [20] and other M/G/1 scheduling theory.

2 PRIOR WORK

Most literature on overheads in queues considers switching between *classes* of jobs [15, 21] or between *queues* in polling systems [4–6]. These models do not capture preemption overhead, as the policies studied do not switch classes or queues *during* a job's service. The only exception we are aware of is Cao and Xie [6], who study a preempt-restart model in which preempted jobs lose all progress, whereas we study a preempt-resume model.

Overhead from preempting individual jobs has not been widely studied. Goerg [13] analyzes SRPT with deterministic pause-only overhead, deriving only mean response time. Peng [16] studies mean response time under preemptive priority with a more flexible overhead model, which we adopt (Section 4). Our work extends theirs by analyzing the *full* response time distribution.

Preemption overhead in preemptive-priority systems introduces a dependency between a job's service length and the number of interruptions it experiences from higher-priority arrivals. Several prior works study queueing systems where the arrival process and service time are interdependent. These include:

- Preempt-repeat systems, where arrivals interrupt and restart service [1–3, 7, 8, 10, 12, 17].
- Multiclass systems where arrival rates vary based on the class of the job in service [9].

These systems differ in the form of arrival-service dependency: in preempt-repeat models, arrivals increase service time directly; in Ernst et al. [9], the job in service alters the arrival process. Most analyses rely on multitype Galton-Watson branching processes, though techniques vary.

Our work introduces a unifying technique: the *job joint transform*, which captures the joint distribution of a job's service time and the arrivals during that time. This tool handles all the above forms of arrival-service dependency and directly yields our stability condition (Theorem 3.1) and busy period results (Theorem 3.2). We have derived job joint transforms for these and other systems; though omitted here for space, they can be used directly with our stability and busy period results.

3 THE JOB JOINT TRANSFORM AND STABILITY CONDITIONS

In many M/G/1 systems, each job has some service time R, and some number of arrivals A occur during that service. In classical models, these quantities are independent: the arrival process proceeds at a fixed rate λ, and conditional on R, we have $A \sim \text{Poisson}(\lambda R)$. That is, the arrival process is independent of the job in service, and the service time does not depend on the number or type of arrivals.

In more complex systems, this assumption may break. For example, the arrival rate may vary depending on the job in service, or a job's service time may increase when certain arrivals occur (e.g., due to interruptions or overhead). We refer to this general phenomenon as *arrival-service dependency*, where the distribution of A depends on R and vice versa. Such systems often involve multiple classes of jobs, where the impact of an arrival may depend on both the class of the arriving job and the class of the job in service. In these cases, we write R_k for the service time of a class k job, and define $\vec{A}_k = (A_{k,1}, \ldots, A_{k,n})$, where $A_{k,i}$ counts the number of class i arrivals that occur during service of a class k job. The joint distribution $(R_k, \vec{A}_k)$ may vary arbitrarily across classes and need not admit any independence structure.

Nonetheless, one can still show that for each class k, the joint distribution $(R_k, \vec{A}_k)$ is drawn i.i.d. across jobs. This follows from the homogeneity of Poisson arrivals, even in the presence of arrival-service dependency. We omit the proof for space.

We therefore define the *job joint transform*, which captures the joint distribution of $(R_k, \vec{A}_k)$ for each class k.

$$\mathcal{J}_k(\theta, \vec{z}) = \mathbb{E}\left[e^{-\theta R_k} \prod_{i=1}^{n} z_i^{A_{k,i}}\right].$$

3.1 Notation

We use the following subscript conventions throughout. A subscript like $< \ell$ denotes a sum over classes $< \ell$ for arrival rates and loads (e.g., $\lambda_{<\ell} = \sum_{i<\ell} \lambda_i$); a class-restricted busy period $B_{<\ell}$ includes only jobs of class $< \ell$ at all recursive levels; and for everything else, $< \ell$ denotes a mixture over classes $< \ell$ weighted by arrival rate; e.g., $z_{<k} = \sum_{i<k} \frac{\lambda_i}{\lambda_{<\ell}} z_i$. Subscripts $\leq \ell$ and $> \ell$ are defined analogously. When no subscript is present, the quantity refers to all classes, and can be interpreted as using $\leq n$. We write $\widetilde{V}$ for the Laplace–Stieltjes transform (LST) of a distribution V, and $\mathcal{E}V$ for its excess distribution [14], whose LST is $\widetilde{\mathcal{E}V}(\theta) = \frac{1-\widetilde{V}(\theta)}{\theta \mathbb{E}[V]}$.

3.2 Stability Conditions

Stability conditions can be directly extracted from the job joint transforms $\mathcal{J}_k(\theta, \vec{z})$. Let $\vec{A}_k = (A_{k,1}, \ldots, A_{k,n})$ denote the vector of class-wise arrivals during a class k job. Then by Wald's equation and multitype Galton-Watson branching process theory, we obtain the following:

Theorem 3.1. *The M/G/1 is stable if*

$$\rho := \sum_{k=1}^{n} \frac{\lambda_k}{\lambda} \sum_{i=1}^{n} \mathbb{E}[A_{k,i}] < 1,$$

i.e., the expected total number of arrivals during a single job's service is less than 1.

In this situation, ρ is also the *load* (a.k.a. utilization) of the system, and can be expressed in several equivalent ways:

$$\rho = \sum_{k=1}^{n} \lambda_k \mathbb{E}[R_k] = \sum_{k=1}^{n} \mathbb{E}[A_{k,k}] \quad \text{and} \quad \rho = -\sum_{k=1}^{n} \lambda_k \partial_\theta \mathcal{J}_k(0, \vec{1}) = \sum_{i=1}^{n} \partial_{z_k} \mathcal{J}_k(0, \vec{1}).$$

The first identity follows from a standard renewal-reward argument [14]; the second expresses these quantities in terms of derivatives of the job joint transforms.

3.3 Busy Period Analysis

A key structural object in our analysis is the *busy period*: the total amount of work generated, directly or indirectly, by a single job or initial period of work. The busy period started by a job includes the service time of that "level 0" job, as well as all recursively generated "level h" jobs for $h \geq 1$, where level h jobs are those that arrive during service of level $h-1$ jobs. We denote by $B(\mathcal{J}_k)$ the transform of the busy period started by a class k job with joint transform $\mathcal{J}_k$, and by $B(V)$ the transform of a busy period started by an arrival-independent amount of work V.

In multiclass systems, it is often useful to restrict attention to jobs of certain classes. We define the *class $< \ell$ busy period* as the total service time recursively generated by jobs of class $< \ell$. That is, only class $< \ell$ jobs are included at all levels of the recursion beyond the initial level 0 job. We denote the transform of the class $< \ell$ busy period started by a class k job as $B_{<\ell}(\mathcal{J}_k)$, and use $B_{<\ell}(V)$ when the initial work is not associated with a specific job. Both of these busy period constructs are central to many response time analyses under preemptive policies [13, 14, 16, 18–20], including our own results in Section 5.

Theorem 3.2.

$$\widetilde{B_{<l}}(\mathcal{J}_k)(\theta) = \mathcal{J}_k\big(\theta, \left[\mathbb{1}(i < l)\, \widetilde{B_{<l}}(\mathcal{J}_i)(\theta) + \mathbb{1}(i \geq l)\right]_{i=1}^{n}\big),$$
$$\widetilde{B_{<l}}(V)(\theta) = \widetilde{V}\big(\theta + \lambda_{<l}(1 - \widetilde{B_{<l}}(\mathcal{J}_{<l})(\theta))\big).$$

4 PREEMPTION OVERHEAD MODEL AND JOB JOINT TRANSFORM

Having obtained the transform for arrival-sensitive busy periods, we will apply it to a system with preemption overhead. We first describe our model and introduce some notation.

We are considering a system with a single server, Poisson arrivals, and *static priority*, meaning each arriving job has a class $k \in \{1, \ldots, n\}$ that remains unchanged during its time in the system. The scheduling policy is *preemptive priority*, meaning that at each point in time, the job with lowest class is being served. If a job of class i arrives during service of a class k job, and $i < k$, the class k job will be preempted to serve the class i job, incurring pause overhead when it is preempted and resume overhead when it resumes service.

We will need notation for the following values for response time analysis. For each class k, λ_k is the arrival rate; S_k, C_k, and D_k denote the base service time (job size), pause overhead, and resume overhead distributions, respectively; and R_k is the full service time including overhead. The quantities γ_k and δ_k denote the loads of pause and resume overheads, respectively. We write $\sigma_k = \lambda_k \mathbb{E}[S_k]$ for the job-size load (service excluding overhead), and $\rho_k = \sigma_k + \gamma_k + \delta_k$ for the total service load including overhead. Finally, as in Section 3.1, we write $B_{<k}(C_i)$ and $B_{<k}(D_i)$ for busy periods started by pause or resume overheads.

As a first step to analyzing preemption overhead, we compute the job joint transform induced by our overhead model.

Theorem 4.1. *The job joint transform for a class k job under our preemption overhead model is*

$$\mathcal{J}_k(\theta, \vec{z}) = \widetilde{S_k}\big(\theta + \lambda_{\geq k}(1 - z_{\geq k}) + \lambda_{<k}\big(1 - z_{<k}\, O_k(\theta, \vec{z})\big)\big), \text{ where}$$

$$O_k(\theta, \vec{z}) = \frac{\widetilde{C_k}(\theta + \lambda(1-z))\,\widetilde{D_k}(\theta + \lambda_{\geq k}(1 - z_{\geq k}) + \lambda_{<k})}{1 - \widetilde{C_k}(\theta + \lambda(1-z))\big(\widetilde{D_k}(\theta + \lambda(1-z)) - \widetilde{D_k}(\theta + \lambda_{\geq k}(1 - z_{\geq k}) + \lambda_{<k})\big)}.$$

From Theorem 4.1, we can compute γ_k and δ_k, obtaining

$$\gamma_k = \frac{\lambda_{<k}\sigma_k}{\widetilde{D_k}(\lambda_{<k})}\mathbb{E}[C_k] \qquad\qquad \delta_k = \frac{\lambda_{<k}\sigma_k}{\widetilde{D_k}(\lambda_{<k})}\mathbb{E}[D_k].$$

5 RESPONSE TIME ANALYSIS

We use the job joint transform $\mathcal{J}$ to analyze response time under preemptive priority with preemption overhead. Given the busy period transforms from Theorem 3.2, we apply a tagged-job style analysis [14, 18, 20].

A class k job must wait behind all class $< k$ jobs present at arrival and any that arrive while it waits. It cannot depart until the full class $< k$ busy period it initiates completes. Thus, its response time corresponds to that of an M/G/1 queue with arrival rate λ_k and job size distribution $B_{<k}(\mathcal{J}_k)$, where these job sizes represent class k jobs' *residence times*: the time from start of service to completion [14].

Additional delays occur if the job arrives during a class $> k$ resume, a class $> k$ pause, a class $> k$ service it interrupts, or a class $< k$ service not yet included in a class k residence time. We model these as *generalized vacations* in the sense of Fuhrmann and Cooper [11], reducing the problem to computing X_k: the response time of class k jobs that do not arrive during another class k job's residence time. The main challenge lies in computing $\widetilde{X_k}(\theta)$.

Theorem 5.1. *The transform of the response time of a class k job under preemption overhead is*

$$\frac{(1 - \rho_{<k} - \rho_k)\widetilde{B_{<k}(\mathcal{J}_k)}(\theta)}{1 - \rho_{<k} - \rho_k \widetilde{\mathcal{E}B_{<k}}(\mathcal{J}_k)(\theta)}\widetilde{X_k}(\theta),$$

where

$$\widetilde{X_k}(\theta) = \sum_{j=k+1}^{n} \frac{\lambda_j}{\lambda_{>k}} \frac{\delta_{>k}}{1-\rho_{\leq k}} \widetilde{\mathcal{E}B_{<k}(D_j)}(\theta)\, \widetilde{B_{<k}(C_j)}(\theta) + \frac{(1-\rho_{<k})\sigma_{>k}}{1-\rho_{\leq k}} \widetilde{B_{<k}(C_{>k})}(\theta)$$
$$+ \sum_{j=k+1}^{n} \sum_{i=1}^{j-1} \frac{\lambda_i}{\lambda_{<j}} \frac{\gamma_j}{1-\rho_{\leq k}} \widetilde{\mathcal{E}B_{<k}}(C_j)(\theta)\, \widetilde{B_{<k}}(\mathcal{J}_i)(\theta)^{\mathbb{1}(i<k)} + \frac{(1-\rho+\sigma_{>k})\rho_{<k}}{1-\rho_{\leq k}} \widetilde{\mathcal{E}B_{<k}(\mathcal{J}_{<k})}(\theta).$$

REFERENCES

[1] Søren Asmussen and Peter W. Glynn. 2017. On Preemptive-Repeat LIFO Queues. *Queueing Systems* 87, 1 (Oct. 2017), 1–22. https://doi.org/10.1007/s11134-017-9532-3

[2] B. Avi-Itzhak. 1963. Preemptive Repeat Priority Queues as a Special Case of the Multipurpose Server Problem—I. *Operations Research* 11, 4 (Aug. 1963), 597–609. https://doi.org/10.1287/opre.11.4.597

[3] Jacob Bergquist and Karl Sigman. 2022. Stationary Workload and Service Times for Some Nonwork-Conserving M/G/1 Preemptive LIFO Queues. *Stochastic Models* 38, 4 (Oct. 2022), 515–544. https://doi.org/10.1080/15326349.2022.2074458

[4] Marko A. A. Boon, Rob D. van der Mei, and Erik M. M. Winands. 2011. Applications of Polling Systems. *Surveys in Operations Research and Management Science* 16, 2 (July 2011), 67–82. https://doi.org/10.1016/j.sorms.2011.01.001

[5] Sem C. Borst and Onno J. Boxma. 2018. Polling: Past, Present, and Perspective. *TOP* 26, 3 (Oct. 2018), 335–369. https://doi.org/10.1007/s11750-018-0484-5

[6] Jianyu Cao and Weixin Xie. 2017. Stability of a Two-Queue Cyclic Polling System with BMAPs under Gated Service and State-Dependent Time-Limited Service Disciplines. *Queueing Systems* 85, 1 (Feb. 2017), 117–147. https://doi.org/10.1007/s11134-016-9504-z

[7] Wei Chang. 1965. Preemptive Priority Queues. *Operations Research* 13, 5 (Oct. 1965), 820–827. https://doi.org/10.1287/opre.13.5.820

[8] Steve Drekic and David A. Stanford. 2001. Reducing Delay in Preemptive Repeat Priority Queues. *Operations Research* 49, 1 (Feb. 2001), 145–156. https://doi.org/10.1287/opre.49.1.145.11186

[9] Philip A. Ernst, Søren Asmussen, and John J. Hasenbein. 2018. Stability and Busy Periods in a Multiclass Queue with State-Dependent Arrival Rates. *Queueing Systems* 90, 3-4 (Dec. 2018), 207–224. https://doi.org/10.1007/s11134-018-9587-9

[10] Val Andrei Fajardo and Steve Drekic. 2017. Waiting Time Distributions in the Preemptive Accumulating Priority Queue. *Methodology and Computing in Applied Probability* 19, 1 (March 2017), 255–284. https://doi.org/10.1007/s11009-015-9476-1

[11] S. W. Fuhrmann and Robert B. Cooper. 1985. Stochastic Decompositions in the $M/G/1$ Queue with Generalized Vacations. *Operations Research* 33, 5 (Oct. 1985), 1117–1129. https://doi.org/10.1287/opre.33.5.1117

[12] Donald P. Gaver. 1962. A Waiting Line with Interrupted Service, Including Priorities. *Journal of the Royal Statistical Society. Series B (Methodological)* 24, 1 (1962), 73–90. http://www.jstor.org/stable/2983746

[13] Carmelita Goerg. 1986. Evaluation of the Optimal SRPT Strategy with Overhead. *IEEE Transactions on Communications* 34, 4 (April 1986), 338–344. https://doi.org/10.1109/TCOM.1986.1096548

[14] Mor Harchol-Balter. 2013. *Performance Modeling and Design of Computer Systems: Queueing Theory in Action.* Cambridge University Press, Cambridge, UK.

[15] Marcel F. Neuts. 1977. The $M/G/1$ Queue with Several Types of Customers and Change-over Times. *Advances in Applied Probability* 9, 3 (Sept. 1977), 604–644. https://doi.org/10.2307/1426117

[16] Edwin Peng. 2022. Exact Response Time Analysis of Preemptive Priority Scheduling with Switching Overhead. *ACM SIGMETRICS Performance Evaluation Review* 49, 2 (Jan. 2022), 72–74. https://doi.org/10.1145/3512798.3512824

[17] Linus Schrage. 1969. A Mixed-Priority Queue with Applications to the Analysis of Real-Time Systems. *Operations Research* 17, 4 (Aug. 1969), 728–742. https://doi.org/10.1287/opre.17.4.728

[18] Linus E. Schrage and Louis W. Miller. 1966. The Queue $M/G/1$ with the Shortest Remaining Processing Time Discipline. *Operations Research* 14, 4 (Aug. 1966), 670–684. https://doi.org/10.1287/opre.14.4.670

[19] Ziv Scully. 2022. *A New Toolbox for Scheduling Theory.* Ph. D. Dissertation. Carnegie Mellon University, Pittsburgh, PA. https://ziv.codes/pdf/scully-thesis.pdf

[20] Ziv Scully, Mor Harchol-Balter, and Alan Scheller-Wolf. 2018. SOAP: One Clean Analysis of All Age-Based Scheduling Policies. *Proceedings of the ACM on Measurement and Analysis of Computing Systems* 2, 1, Article 16 (March 2018), 30 pages. https://doi.org/10.1145/3179419

[21] Mark P. Van Oyen, Dimitrios G. Pandelis, and Demosthenis Teneketzis. 1992. Optimality of Index Policies for Stochastic Scheduling with Switching Penalties. *Journal of Applied Probability* 29, 4 (Dec. 1992), 957–966. https://doi.org/10.2307/3214727