Optimal Scheduling in the Multiserver-job Model under Heavy Traffic

Isaac Grosof
igrosof@cs.cmu.edu
Carnegie Mellon University
Computer Science Department
Pittsburgh, PA, USA

Ziv Scully
zivscully@cornell.edu
Cornell University
School of Operations Research and Information Engineering
Ithaca, NY, USA

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

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

ABSTRACT

Multiserver-job systems, where jobs require concurrent service at many servers, occur widely in practice. Essentially all of the theoretical work on multiserver-job systems focuses on maximizing utilization, with almost nothing known about mean response time. Our goal in this paper is to minimize mean response time in a multiserver-job setting. Minimizing mean response time requires prioritizing small jobs while simultaneously maximizing utilization. Our question is how to achieve these joint objectives.

We devise the ServerFilling-SRPT scheduling policy, which is the first policy to minimize mean response time in the multiserver-job model in the heavy traffic limit. In addition to proving this heavy-traffic result, we present empirical evidence that ServerFilling-SRPT outperforms all existing scheduling policies for all loads, with orders of magnitude improvements at high load.

Because ServerFilling-SRPT requires knowing job sizes, we also define the ServerFilling-Gittins policy, which is optimal when sizes are unknown or partially known.

For more detail, see the full paper, [8].

CCS CONCEPTS

• **General and reference → Performance**; • **Mathematics of computing → Queueing theory**; • **Theory of computation →** *Scheduling algorithms.*

KEYWORDS

scheduling; SRPT; Gittins; multiserver-job; response time; latency; sojurn time; heavy traffic; asymptotic optimality

ACM Reference Format:
Isaac Grosof, Ziv Scully, Mor Harchol-Balter, and Alan Scheller-Wolf. 2023. Optimal Scheduling in the Multiserver-job Model under Heavy Traffic. In *Abstract Proceedings of the 2023 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS '23 Abstracts), June 19–23, 2023, Orlando, FL, USA.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3578338.3593560

1 THE MULTISERVER-JOB MODEL

Traditional multiserver queueing theory focuses on models, such as the M/G/k, where every job occupies exactly one server. For decades, these models remained popular because they captured the behavior of computing systems, while being amenable to theoretical analysis. However, such one-server-per-job models are no longer representative of many modern computing systems.

Consider today's large-scale computing centers, such as the those of Google, Amazon and Microsoft. While the *servers* in these data centers still resemble the *servers* in traditional models such as the M/G/k, the *jobs* have changed: Each job now requires many servers, which it holds simultaneously [9, 10]. The distribution of the number of CPUs requested by jobs in Google's recently published trace of its "Borg" computation cluster [5, 17] is highly variable, with jobs requesting anywhere from 1 to 100,000 normalized CPUs. We focus on this "multiserver-job model" (MSJ), by which we refer to the common situation in modern systems where each job concurrently occupies a fixed number of servers (typically more than one), throughout its time in service.

The multiserver-job model is fundamentally different from the one-server-per-job model. In the one-server-per-job model, any work-conserving scheduling policy such as First-Come First-Served (FCFS) can achieve full server utilization. By contrast, in the multiserver-job model, a naïve scheduling policy such as FCFS will waste more servers than necessary. As a result, server utilization and system stability are dependent on the scheduling policy in the multiserver-job model. While finding throughput-optimal scheduling policies is a challenge, several such policies are known, including MaxWeight [11], Randomized Timers [3, 12], and ServerFilling [5]. Among these, mean response time is only understood for ServerFilling [5], and minimizing the mean response time has never been a goal of this line of work.

SIGMETRICS '23 Abstracts, June 19–23, 2023, Orlando, FL, USA.

ACM ISBN 979-8-4007-0074-3/23/06.
https://doi.org/10.1145/3578338.3593560

2 CHALLENGES OF MINIMIZING MSJ MEAN RESPONSE TIME

In the M/G/k setting, where each job requires a single server, it was recently proven that the SRPT-k (Shortest Remaining Processing Time-k) scheduling policy minimizes mean response time in the heavy-traffic limit [6]. SRPT-k is a very simple policy: serve the k jobs of least remaining duration (service time).

Unfortunately, trying to simply adapt the SRPT-k policy does not result in an optimal policy for two reasons:

- Prioritizing by remaining job duration is not the right approach. Instead, we must focus on *size*, the product of duration and the number of servers required.
- Greedily prioritizing the job of least remaining size, as in SRPT-k, is not throughput optimal. Our policy must be throughput-optimal, while *also* prioritizing small jobs.

We therefore ask:

> *What scheduling policy for the multiserver-job model should we use to* minimize mean response time *in the heavy-traffic limit?*

By "heavy-traffic" we mean as load $\rho \to 1$, while the number of servers, k, stays fixed.

3 SERVERFILLING-SRPT AND GENERALIZATIONS

We introduce the ServerFilling-SRPT scheduling policy, the first scheduling policy to minimize mean response time in the multiserver-job model in the heavy traffic limit.

ServerFilling-SRPT is defined in the setting where k is a power of 2, and all server needs are powers of 2. This setting is commonly seen in practice in supercomputing and other highly-parallel computing settings [1, 2].

To define ServerFilling-SRPT, imagine all jobs are ordered by their remaining size. Select the smallest initial subset M of this sequence such that the jobs in M collectively require at least k servers. Finally, place jobs from M into service in order of largest server need. This procedure is performed preemptively, whenever a job arrives or completes. Using the fact that all servers needs are powers of 2, and k is a power of 2, we prove that whenever jobs with total server need at least k are present in the system, this procedure will fill all k servers. We use this property to prove that ServerFilling-SRPT minimizes mean response time in the heavy-traffic limit.

ServerFilling-SRPT requires the scheduler to know job durations, and hence sizes, in advance. Sometimes the scheduler does not have duration information. In the M/G/1 setting, when job sizes are unknown, the Gittins policy [4] is known to achieve optimal mean response time. We therefore introduce the ServerFilling-Gittins policy, and prove similar heavy-traffic optimality results for it.

While ServerFilling-SRPT requires that the server needs are powers of 2, we have developed a more general scheduling policy which requires only that the server needs all divide k. We call this generalization DivisorFilling-SRPT. We then show that all of our results about ServerFilling-SRPT and ServerFilling-Gittins hold for DivisorFilling-SRPT and DivisorFilling-Gittins.

4 A NOVEL PROOF TECHNIQUE: MIAOW

In recent years, there have been a plethora of proof techniques developed to handle the analysis of multiserver systems. These include:

- Multiserver tagged job analysis [6, 7, 16],
- Worst-case work gap [6, 7, 16],
- WINE (Work Integral Number Equality) [14, Chapter 4] [13, 15],
- Work Decomposition law [15].

While many of these techniques are used in this paper, they do not *suffice* to handle the analysis of ServerFilling-SRPT. The analysis of ServerFilling-SRPT hinges on bounding the *waste* relative to a resource-pooled single-server SRPT system, where waste is the expected product of work and unused system capacity. In order to analyze waste, we introduce a new technique called MIAOW, Multiplicative Interval Analysis of Waste. MIAOW buckets jobs into multiplicative intervals based on their remaining sizes, and bounds the waste in each interval.

REFERENCES

[1] Walfredo Cirne and Francine Berman. 2001. A model for moldable supercomputer jobs. In *Proceedings 15th International Parallel and Distributed Processing Symposium. IPDPS 2001.* 8 pp.

[2] Allen B. Downey. 1997. Using queue time predictions for processor allocation. In *workshop on Job Scheduling Strategies for Parallel Processing*. Springer, 35–57.

[3] Javad Ghaderi. 2016. Randomized algorithms for scheduling VMs in the cloud. In *IEEE INFOCOM 2016 - The 35th Annual IEEE International Conference on Computer Communications.* 1–9.

[4] John Gittins, Kevin Glazebrook, and Richard Weber. 2011. *Multi-armed bandit allocation indices.* John Wiley & Sons.

[5] Isaac Grosof, Mor Harchol-Balter, and Alan Scheller-Wolf. 2022. WCFS: A new framework for analyzing multiserver systems. *Queueing Systems* (2022).

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

[7] Isaac Grosof, Ziv Scully, and Mor Harchol-Balter. 2019. Load Balancing Guardrails: Keeping Your Heavy Traffic on the Road to Low Response Times. *Proc. ACM Meas. Anal. Comput. Syst.* 3, 2, Article 42 (jun 2019), 31 pages.

[8] Isaac Grosof, Ziv Scully, Mor Harchol-Balter, and Alan Scheller-Wolf. 2022. Optimal Scheduling in the Multiserver-job Model under Heavy Traffic. *Proceedings of the ACM on Measurement and Analysis of Computing Systems* 6, 3 (2022), 1–32.

[9] Mor Harchol-Balter. 2021. Open problems in queueing theory inspired by datacenter computing. *Queueing Systems: Theory and Applications* 97, 1 (2021), 3–37.

[10] Mor Harchol-Balter. 2022. The multiserver job queueing model. *Queueing Systems* 100, 3 (2022), 201–203.

[11] Siva Theja Maguluri, Rayadurgam Srikant, and Lei Ying. 2012. Stochastic models of load balancing and scheduling in cloud computing clusters. In *2012 Proceedings IEEE Infocom.* IEEE, 702–710.

[12] Konstantinos Psychas and Javad Ghaderi. 2018. Randomized Algorithms for Scheduling Multi-Resource Jobs in the Cloud. *IEEE/ACM Transactions on Networking* 26, 5 (2018), 2202–2215.

[13] Ziv Scully. 2021. WINE: A New Queueing Identity for Analyzing Scheduling Policies in Multiserver Systems. https://ziv.codes/pdf/wine-talk.pdf INFORMS Annual Meeting.

[14] Ziv Scully. 2022. *A New Toolbox for Scheduling Theory.* Ph. D. Dissertation. Carnegie Mellon University.

[15] 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 (Nov. 2020), 29 pages.

[16] Ziv Scully, Isaac Grosof, and Mor Harchol-Balter. 2021. Optimal multiserver scheduling with unknown job sizes in heavy traffic. *Performance Evaluation* 145 (2021), 102150.

[17] Muhammad Tirmazi, Adam Barker, Nan Deng, Md E. Haque, Zhijing Gene Qin, Steven Hand, Mor Harchol-Balter, and John Wilkes. 2020. Borg: The next Generation. In *Proceedings of the Fifteenth European Conference on Computer Systems* (Heraklion, Greece) *(EuroSys '20).* Association for Computing Machinery, New York, NY, USA, Article 30, 14 pages.