Size-Based Scheduling in Service Systems

Rouba Ibrahim
University College London

Joint work with:

Jing Dong
Columbia Business School

LNMB Conference 2024

In a nutshell

We study scheduling policies based on (noisy) service-time information in service systems.

A motivational example: Scheduling in a bank's call center

Data set from SEE Lab (Technion).

- Individual-level call data (April 2007 - June 2009)
- We can track customers using their unique ID's
- Callers contact 12 times on average
- 1,835 agents in total, 400-450 agents on a day
- Customer abandonment: 4.5%

A motivational example: Scheduling in a bank's call center

Data set from SEE Lab (Technion).

- Individual-level call data (April 2007 - June 2009)
- We can track customers using their unique ID's
- Callers contact 12 times on average
- 1,835 agents in total, 400-450 agents on a day
- Customer abandonment: 4.5%

Can we use data from the past transactions of a customer to predict their service times, and use these predictions to schedule more efficiently?

Size-based scheduling

If service times are perfectly known and preemptions are allowed, then schedule the Shortest Remaining Processing Time first.

Size-based scheduling

If service times are perfectly known and preemptions are allowed, then schedule the Shortest Remaining Processing Time first.

SRPT minimizes the mean response time in $M/G/1$.

(Schrage and Miller 1966; Wierman 2008; Harchol-Balter 2013; ...)

Size-based scheduling

If service times are perfectly known and preemptions are allowed, then schedule the Shortest Remaining Processing Time first.

SRPT minimizes the mean response time in $M/G/1$.

(Schrage and Miller 1966; Wierman 2008; Harchol-Balter 2013; ...)

SRPT is asymptotically optimal in heavy traffic in $M/G/k$.

(Grosof, Scully, Harchol-Balter 2018.)

Size-based scheduling

If service times are perfectly known and preemptions are allowed, then schedule the Shortest Remaining Processing Time first.

SRPT minimizes the mean response time in $M/G/1$.

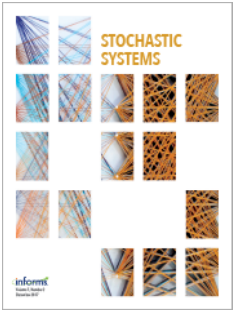
(Schrage and Miller 1966; Wierman 2008; Harchol-Balter 2013; ...)

SRPT is asymptotically optimal in heavy traffic in $M/G/k$.

(Grosof, Scully, Harchol-Balter 2018.)

What if we have a noisy estimate of the service time, and the model is $M/GI/k + GI$?

A difficult problem, even in $M/G/1$



Home > Stochastic Systems > Vol. 9, No. 3 >

Open Problem—Size-Based Scheduling with Estimation Errors

Douglas G. Down

Published Online: 18 Sep 2019 | https://doi.org/10.1287/stsy.2019.0041

Lu D, Sheng H, Dinda P (2004); Wierman and Nuyens (2008); Dell'Amico M, Carra D, Pastorelli M, Michiardi P (2014); Mailach and Down (2017); Scully, Grosof and Harchol-Balter (2020); Mitzenmacher (2021); Scully and Harchol-Balter (2021); Scully, Grosof, and Mitzenmacher (2022); Chen and Dong (2022).

Customer abandonment

Customer abandonment

Assume that we know the service times perfectly (a priori).

Customer abandonment

Assume that we know the service times perfectly (a priori).

Model the service system as a multiserver queue with abandonment.

Customer abandonment

Assume that we know the service times perfectly (a priori).

Model the service system as a multiserver queue with abandonment.

In $M/G/1$, we know that SRPT is optimal.

Customer abandonment

Assume that we know the service times perfectly (a priori).

Model the service system as a multiserver queue with abandonment.

In $M/G/1$, we know that SRPT is optimal.

In $M/G/k$, we know that SRPT is asymptotically optimal.

Customer abandonment

Assume that we know the service times perfectly (a priori).

Model the service system as a multiserver queue with abandonment.

In $M/G/1$, we know that SRPT is optimal.

In $M/G/k$, we know that SRPT is asymptotically optimal.

Question: Assuming service times are perfectly known, what do we know about SRPT in multi-server queues with abandonment?

Customer abandonment

Assume that we know the service times perfectly (a priori).

Model the service system as a multiserver queue with abandonment.

In $M/G/1$, we know that SRPT is optimal.

In $M/G/k$, we know that SRPT is asymptotically optimal.

Question: Assuming service times are perfectly known, what do we know about SRPT in multi-server queues with abandonment?

Answer: Nothing.

Plan for the talk

1. Service times perfectly known: SRPT in $M/GI/s + GI$

   Dong and Ibrahim. 2021. *SRPT scheduling in many-server queues with impatient customers.* Management Science.

2. Noisy service-time information: SJF in $M/GI/s + GI$

   Dong and Ibrahim. 2023. *SJF scheduling in many-server queues with impatient customers and noisy service-time estimates.*

Perfect Service-Time Information: SRPT in $M/GI/s + GI$

Snapshot of main results

Here, we will consider an asymptotic many-server overloaded regime, and we will focus on throughput.

(Atar, Giat, Shimkin 2010; Atar, Kaspi, Shimkin 2014; Bassamboo and Randhawa 2016; Puha and Ward 2021; ...)

Snapshot of main results

Here, we will consider an asymptotic many-server overloaded regime, and we will focus on throughput.

(Atar, Giat, Shimkin 2010; Atar, Kaspi, Shimkin 2014; Bassamboo and Randhawa 2016; Puha and Ward 2021; ...)

We will derive limits for steady-state performance measures.

Snapshot of main results

Here, we will consider an asymptotic many-server overloaded regime, and we will focus on throughput.

(Atar, Giat, Shimkin 2010; Atar, Kaspi, Shimkin 2014; Bassamboo and Randhawa 2016; Puha and Ward 2021; ...)

We will derive limits for steady-state performance measures.

We will demonstrate that SRPT asymptotically maximizes the throughput.

Snapshot of main results

Here, we will consider an asymptotic many-server overloaded regime, and we will focus on throughput.

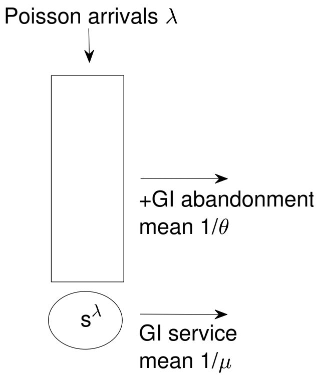
(Atar, Giat, Shimkin 2010; Atar, Kaspi, Shimkin 2014; Bassamboo and Randhawa 2016; Puha and Ward 2021; ...)

We will derive limits for steady-state performance measures.

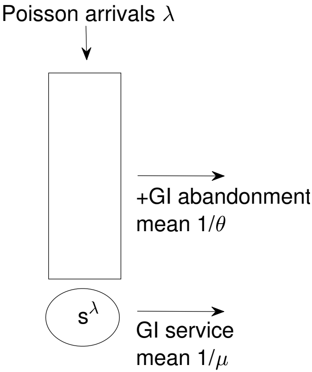
We will demonstrate that SRPT asymptotically maximizes the throughput.

We will show that the SRPT system is well-approximated asymptotically by a two-priority system.

Modelling framework



Modelling framework



- Keep traffic intensity $\rho = \lambda / s^\lambda \mu > 1$ fixed
- Made stable by abandonments
- Let $\lambda \uparrow \infty$ and $s^\lambda \uparrow \infty$
- Abandonment and service-time distributions fixed
- Non-negligible abandonment/delay in the limit

SRPT scheduling

SRPT scheduling

- Preemptions are allowed.

SRPT scheduling

- Preemptions are allowed.
- Arrival who finds empty server: begins service immediately.

SRPT scheduling

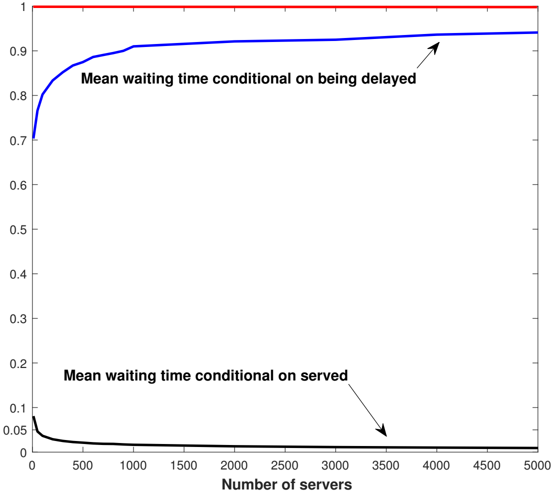
- Preemptions are allowed.
- Arrival who finds empty server: begins service immediately.
- If all servers busy:
  - Update remaining processing times of all jobs in service
  - If service time of arrival $<$ longest remaining processing time in service $\Rightarrow$ preempt the longest remaining processing time
  - Else, join queue.

Simulation results under SRPT

We consider the $M/M/s + E_2$ system with $\rho = 1.4$ and $1/\theta = 1$.

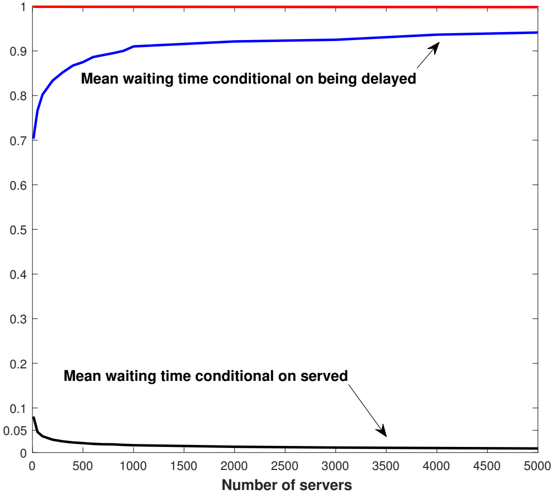
Simulation results under SRPT

We consider the $M/M/s + E_2$ system with $\rho = 1.4$ and $1/\theta = 1$.



Simulation results under SRPT

We consider the $M/M/s + E_2$ system with $\rho = 1.4$ and $1/\theta = 1$.



State-space collapse.

Threshold

Define the threshold τ:

$$\lambda \cdot \mathbb{P}(S \leq \tau) \cdot \mathbb{E}[S|S \leq \tau] = \lambda \mathbb{E}[S\mathbf{1}(S \leq \tau)] = s,$$

where we have:

- λ: arrival rate
- S: service time
- s: number of servers

See Chen and Dong (2022) for a similar idea in $GI/GI/1$.

Main Theorem

For the sequence of $M/GI/s^{\lambda} + GI$ queues under SRPT with $\rho^{\lambda} = \lambda/s_{\lambda}\mu > 1$ held fixed, in steady state:

1. $\lim_{\lambda\to\infty} \mathbb{P}(Served|S \leq \tau) = 1$.
2. $\lim_{\lambda\to\infty} \mathbb{P}(Served|S > \tau) = 0$.
3. $\lim_{\lambda\to\infty} \mathbb{E}[W|Served] = 0$.
4. $\lim_{\lambda\to\infty} \mathbb{E}[W|Abandon] =$ Mean time to abandon.
5. SRPT maximizes throughput.

Main Theorem

For the sequence of $M/GI/s^{\lambda} + GI$ queues under SRPT with $\rho^{\lambda} = \lambda / s_{\lambda}\mu > 1$ held fixed, in steady state:

1. $\lim_{\lambda \to \infty} \mathbb{P}(Served | S \leq \tau) = 1.$
2. $\lim_{\lambda \to \infty} \mathbb{P}(Served | S > \tau) = 0.$
3. $\lim_{\lambda \to \infty} \mathbb{E}[W | Served] = 0.$
4. $\lim_{\lambda \to \infty} \mathbb{E}[W | Abandon] =$ Mean time to abandon.
5. SRPT maximizes throughput.

That is, customers with short service times (below threshold) are served immediately, and customers with long service times eventually abandon.

Proof idea

Proof idea

It is hard to prove this directly.

Proof idea

It is hard to prove this directly.

Customers with long service times abandon.

Proof idea

It is hard to prove this directly.

Customers with long service times abandon.

Customers with short service times are served immediately.

Proof idea

It is hard to prove this directly.

Customers with long service times abandon.

Customers with short service times are served immediately.

This looks like fluid performance in a large queue with two priority classes, where the class is defined according to the service time.

Proof idea

It is hard to prove this directly.

Customers with long service times abandon.

Customers with short service times are served immediately.

This looks like fluid performance in a large queue with two priority classes, where the class is defined according to the service time.

Use a coupling proof with a loss queue with two priority classes.

Sketch of the proof

Sketch of the proof

Consider a loss $M/GI/s/s$ system where class 1 customers ($S < \tau$) have preemptive priority over class 2 customers ($S \geq \tau$).

Sketch of the proof

Consider a loss $M/GI/s/s$ system where class 1 customers ($S < \tau$) have preemptive priority over class 2 customers ($S \geq \tau$).

Coupling proof:

Sketch of the proof

Consider a loss $M/GI/s/s$ system where class 1 customers ($S < \tau$) have preemptive priority over class 2 customers ($S \geq \tau$).

Coupling proof:

1. Couple arrival and service times between loss and SRPT systems. Initialize both systems as empty.

Sketch of the proof

Consider a loss $M/GI/s/s$ system where class 1 customers ($S < \tau$) have preemptive priority over class 2 customers ($S \geq \tau$).

Coupling proof:

1. Couple arrival and service times between loss and SRPT systems. Initialize both systems as empty.
2. Match each class 1 customer in loss system with a customer in service in SRPT system who finishes service earlier.

Sketch of the proof

Consider a loss $M/GI/s/s$ system where class 1 customers ($S < \tau$) have preemptive priority over class 2 customers ($S \geq \tau$).

Coupling proof:

1. Couple arrival and service times between loss and SRPT systems. Initialize both systems as empty.
2. Match each class 1 customer in loss system with a customer in service in SRPT system who finishes service earlier.
3. Conclude, by induction, that you serve more customers in the SRPT system.

Sketch of the proof

Consider a loss $M/GI/s/s$ system where class 1 customers ($S < \tau$) have preemptive priority over class 2 customers ($S \geq \tau$).

Coupling proof:

1. Couple arrival and service times between loss and SRPT systems. Initialize both systems as empty.
2. Match each class 1 customer in loss system with a customer in service in SRPT system who finishes service earlier.
3. Conclude, by induction, that you serve more customers in the SRPT system.
4. In loss system, all class 1 customers are served asymptotically and throughput is maximal.

Sketch of the proof

Consider a loss $M/GI/s/s$ system where class 1 customers ($S < \tau$) have preemptive priority over class 2 customers ($S \geq \tau$).

Coupling proof:

1. Couple arrival and service times between loss and SRPT systems. Initialize both systems as empty.
2. Match each class 1 customer in loss system with a customer in service in SRPT system who finishes service earlier.
3. Conclude, by induction, that you serve more customers in the SRPT system.
4. In loss system, all class 1 customers are served asymptotically and throughput is maximal.
5. Conclude that throughput is maximal in SRPT system, and derive limits for remaining performance measures.

Implications

We show that, in the overloaded $M/GI/s^{\lambda} + GI$ as $\lambda \uparrow \infty$:

Implications

We show that, in the overloaded $M/GI/s^{\lambda} + GI$ as $\lambda \uparrow \infty$:

- SRPT scheduling maximizes throughput among all scheduling policies.

Implications

We show that, in the overloaded $M/GI/s^{\lambda} + GI$ as $\lambda \uparrow \infty$:

- SRPT scheduling maximizes throughput among all scheduling policies.
- SRPT minimizes the expected waiting time conditional on being served.

Implications

We show that, in the overloaded $M/GI/s^{\lambda} + GI$ as $\lambda \uparrow \infty$:

- SRPT scheduling maximizes throughput among all scheduling policies.
- SRPT minimizes the expected waiting time conditional on being served.
- SRPT maximizes the expected waiting time conditional on abandoning.

Implications

We show that, in the overloaded $M/GI/s^{\lambda} + GI$ as $\lambda \uparrow \infty$:

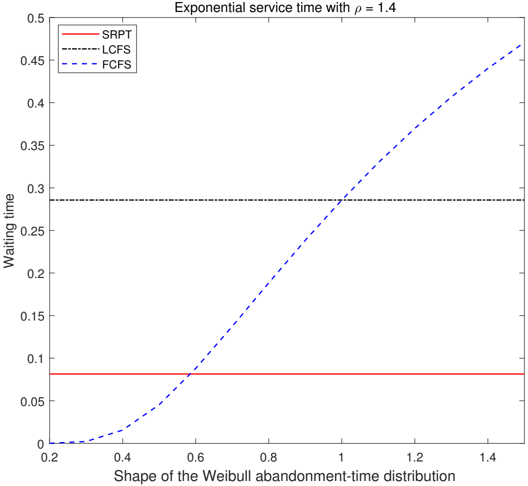
- SRPT scheduling maximizes throughput among all scheduling policies.
- SRPT minimizes the expected waiting time conditional on being served.
- SRPT maximizes the expected waiting time conditional on abandoning.

Implications

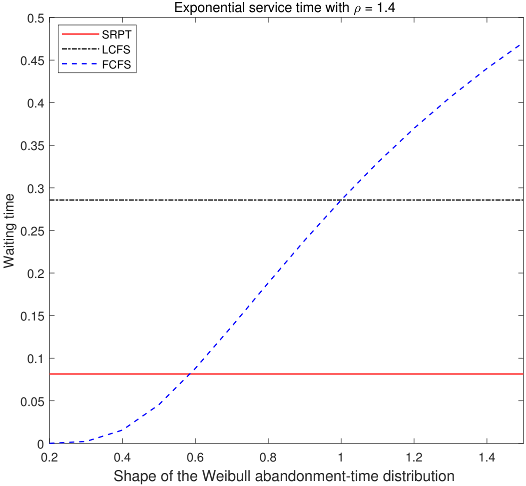
We show that, in the overloaded $M/GI/s^{\lambda} + GI$ as $\lambda \uparrow \infty$:

- SRPT scheduling maximizes throughput among all scheduling policies.
- SRPT minimizes the expected waiting time conditional on being served.
- SRPT maximizes the expected waiting time conditional on abandoning.
- Performance under SRPT is insensitive to the abandonment distribution, beyond the mean.

Effect of the abandonment distribution

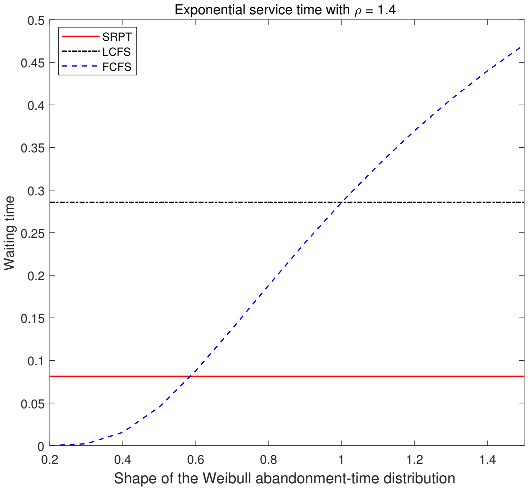


Effect of the abandonment distribution



Recall that, among blind policies:

Effect of the abandonment distribution

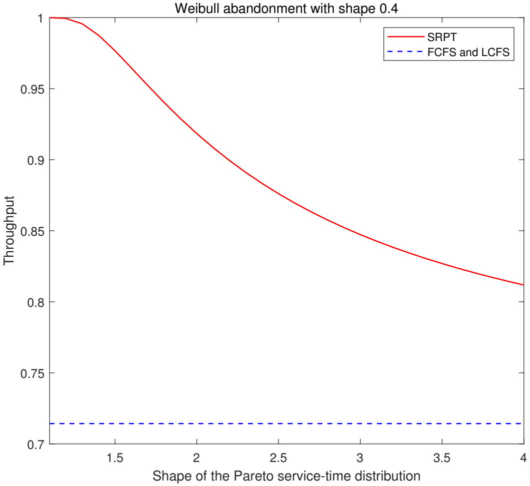


Recall that, among blind policies:

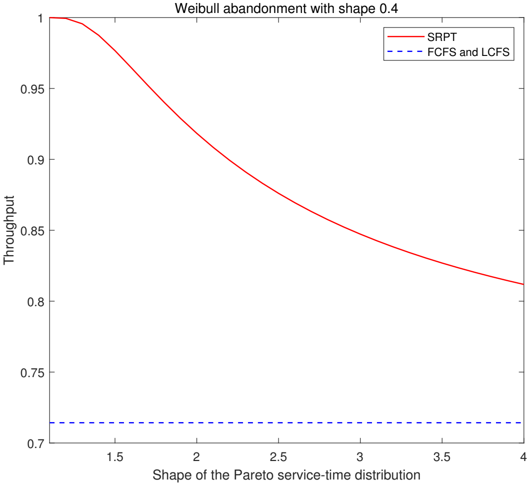
- Weibull shape $\alpha < 1 \Rightarrow$ DHR $\Rightarrow$ FCFS minimizes fluid waiting time
- Weibull shape $\alpha > 1 \Rightarrow$ IHR $\Rightarrow$ LCFS minimizes fluid waiting time

Bassamboo and Randhawa (2015).

Effect of the service-time distribution



Effect of the service-time distribution



- $\uparrow$ shape Pareto service time $\Rightarrow$ lighter tail
- SRPT has stronger advantage under heavier tails

Lin, Wierman, Zwart (2011), Chen and Dong (2022).

Asymptotically, SRPT maximizes throughput and performs like a two-class preemptive priority queue.

Asymptotically, SRPT maximizes throughput and performs like a two-class preemptive priority queue.

What if service-time predictions are noisy?

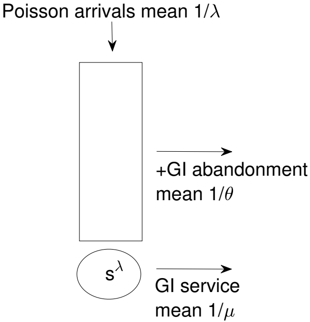
Asymptotically, SRPT maximizes throughput and performs like a two-class preemptive priority queue.

What if service-time predictions are noisy?

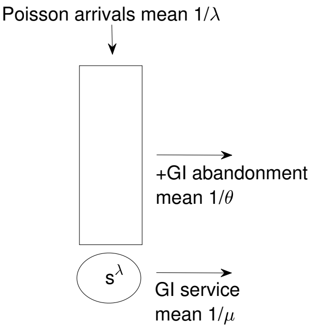
What if preemptions are not allowed?

Imperfect Service-Time Information: SJF in $M/GI/s + GI$

Modelling framework



Modelling framework



- Keep traffic intensity $\rho = \lambda / s^\lambda \mu > 1$ fixed
- Made stable by abandonments
- Let $\lambda \uparrow \infty$ and $s^\lambda \uparrow \infty$
- Abandonment and service-time distributions fixed
- Non-negligible abandonment/delay in the limit

SJF scheduling with service-time predictions

SJF scheduling with service-time predictions

- Preemptions are not allowed.
- Arrival who finds empty server: begins service immediately.
- If all servers busy, join queue.
- When there is a service completion, schedule shortest predicted service time from queue first.

Imperfect service-time information

Let S_i denote the actual service time and $\hat{S}_i$ denote the predicted service time for customer i.

Imperfect service-time information

Let S_i denote the actual service time and $\hat{S}_i$ denote the predicted service time for customer i.

We assume that $\mathbb{E}[S_i|\hat{S}_i = y]$ increases in y, for any y.

Imperfect service-time information

Let S_i denote the actual service time and $\hat{S}_i$ denote the predicted service time for customer i.

We assume that $\mathbb{E}[S_i|\hat{S}_i = y]$ increases in y, for any y.

For example, this is satisfied in a regression model:

$$S_i = \hat{S}_i + \epsilon_i,$$

where $\hat{S}_i$ and ϵ_i are independent.

An updated threshold

Recall how we defined the threshold τ earlier:

$$\lambda \cdot \mathbb{P}(S \leq \tau) \cdot \mathbb{E}[S | S \leq \tau] = \lambda \mathbb{E}[S \mathbf{1}(S \leq \tau)] = s.$$

An updated threshold

Recall how we defined the threshold τ earlier:

$$\lambda \cdot \mathbb{P}(S \leq \tau) \cdot \mathbb{E}[S | S \leq \tau] = \lambda \mathbb{E}[S \mathbf{1}(S \leq \tau)] = s.$$

Now, we define the threshold $\hat{\tau}$ as follows:

$$\lambda \cdot \mathbb{P}(\hat{S} \leq \hat{\tau}) \cdot \mathbb{E}[S | \hat{S} \leq \hat{\tau}] = \lambda \mathbb{E}[S \mathbf{1}(\hat{S} \leq \hat{\tau})] = s.$$

An updated threshold

Recall how we defined the threshold τ earlier:

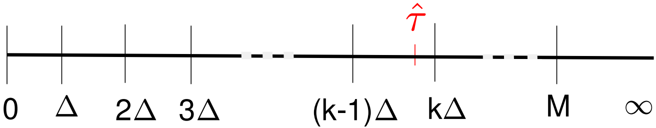
$$\lambda \cdot \mathbb{P}(S \leq \tau) \cdot \mathbb{E}[S | S \leq \tau] = \lambda \mathbb{E}[S \mathbf{1}(S \leq \tau)] = s.$$

Now, we define the threshold $\hat{\tau}$ as follows:

$$\lambda \cdot \mathbb{P}(\hat{S} \leq \hat{\tau}) \cdot \mathbb{E}[S | \hat{S} \leq \hat{\tau}] = \lambda \mathbb{E}[S \mathbf{1}(\hat{S} \leq \hat{\tau})] = s.$$

We show that, asymptotically, prioritizing customers with $\hat{S} < \hat{\tau}$ over customers with $\hat{S} \geq \hat{\tau}$ maximizes the throughput.

"Discretized" SJF Policy: SJF^{Δ}



- Class 1: $\hat{S} \in [0, \Delta)$
- Class 2: $\hat{S} \in [\Delta, 2\Delta)$
- Class 3: $\hat{S} \in [2\Delta, 3\Delta)$
- $\ldots$
- Class k: $\hat{S} \in [(k-1)\Delta, k\Delta)$
- $\ldots$
- Class $\lfloor M/\Delta \rfloor$: $\hat{S} \in [(\lfloor M/\Delta \rfloor - 1)\Delta, M)$
- Class $\lfloor M/\Delta \rfloor + 1$: $\hat{S} \in [M, \infty)$

Class has lower index $\Rightarrow$ Higher non-preemptive priority.

Main Theorem

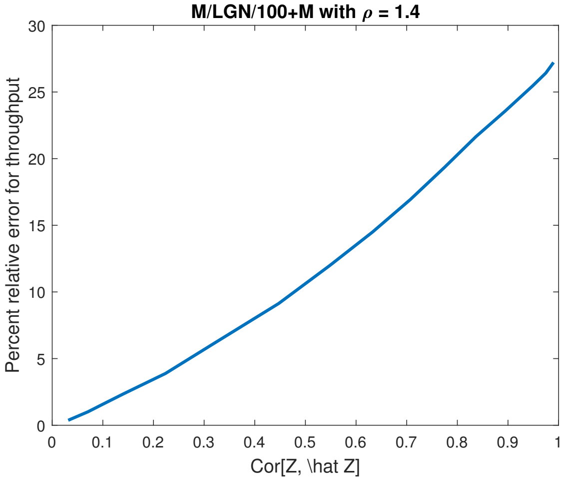
For the sequence of $M/GI/s^{\lambda} + GI$ queues under SJF^{Δ} with $\rho^{\lambda} = \lambda / s_{\lambda}\mu > 1$ held fixed, in steady state:

1. $\lim_{\Delta \downarrow 0} \lim_{\lambda \to \infty} \mathbb{P}(\mathit{Served} | \hat{S} \leq \hat{\tau}) = 1.$
2. $\lim_{\Delta \downarrow 0} \lim_{\lambda \to \infty} \mathbb{P}(\mathit{Served} | \hat{S} > \hat{\tau}) = 0.$
3. $\lim_{\Delta \downarrow 0} \lim_{\lambda \to \infty} \mathbb{E}[W | \mathit{Served}] = 0.$
4. $\lim_{\Delta \downarrow 0} \lim_{\lambda \to \infty} \mathbb{E}[W | \mathit{Abandon}] =$ Mean time to abandon.
5. SJF^{Δ} maximizes throughput among non-preemptive policies that use the noisy service-time information.
6. SJF^{Δ} and SJF have asymptotically the same performance.

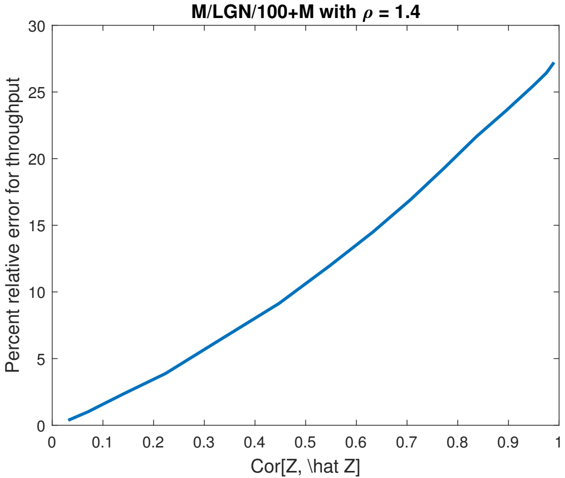
Use Atar, Kaspi, Shimkin (2014).

Accuracy of the approximation: SJF vs. Two-class priority rule

Accuracy of the approximation: SJF vs. Two-class priority rule



Accuracy of the approximation: SJF vs. Two-class priority rule



- The noisier the service times, the better the two-class approximation
- Recall that service-time predictions can be very noisy

Number of classes: Effect on throughput

Number of classes: Effect on throughput

Consider the $M/LGN/100+M$ with $\rho = 1.4$.

Number of classes: Effect on throughput

Consider the $M/LGN/100 + M$ with $\rho = 1.4$.

Divide the high class ($< \tau$) into equally-sized classes.

Number of classes: Effect on throughput

Consider the $M/LGN/100 + M$ with $\rho = 1.4$.

Divide the high class ($< \tau$) into equally-sized classes.

$r[Z, \hat{Z}]$	SJF	2 classes	3 classes	5 classes	10 classes
0.03	0.723	0.722	0.722	0.722	0.722
0.2	0.7677	0.7587	0.7477	0.7657	0.7674
0.5	0.8286	0.807	0.8201	0.8239	0.8243
0.7	0.8593	0.8314	0.8472	0.8487	0.8488
0.95	0.8764	0.8449	0.8595	0.8605	0.8606
0.99	0.8802	0.8474	0.8618	0.8626	0.8626

Number of classes: Effect on throughput

Consider the $M/LGN/100 + M$ with $\rho = 1.4$.

Divide the high class ($< \tau$) into equally-sized classes.

$r[Z, \hat{Z}]$	SJF	2 classes	3 classes	5 classes	10 classes
0.03	0.723	0.722	0.722	0.722	0.722
0.2	0.7677	0.7587	0.7477	0.7657	0.7674
0.5	0.8286	0.807	0.8201	0.8239	0.8243
0.7	0.8593	0.8314	0.8472	0.8487	0.8488
0.95	0.8764	0.8449	0.8595	0.8605	0.8606
0.99	0.8802	0.8474	0.8618	0.8626	0.8626

- low correlation $\Rightarrow$ two-priority approximation very accurate
- high correlation $\Rightarrow$ some advantage in using 3 classes instead

Takeaways

- Theoretical results about performance of SRPT and SJF in many-server queues with abandonment.
- Implementing SRPT or SJF is hard. Usually, only two or three classes sufficient.
- Accuracy of approximation improves as the noise in the service-time prediction increases.

Thank you!