Proceedings of the 2021 Winter Simulation Conference
S. Kim, B. Feng, K. Smith, S. Masoud, Z. Zheng, C. Szabo, and M. Loper, eds.

COMBINING SIMULATION AND MACHINE LEARNING FOR RESPONSE TIME PREDICTION FOR THE SHORTEST REMAINING PROCESSING TIME DISCIPLINE

Jamol Pender
Sukriti Sudhakar
Eva Zhang

Operations Research and Information Engineering
Cornell University
Ithaca, NY 14853, USA

ABSTRACT

Data centers have become a large part of our world and infrastructure and now many of them want to provided information to their customers about the response times of their jobs. To this end, it is important to be able to predict what the response times of jobs might be when a data center implements a specific queueing discipline. In this paper, we investigate the feasibility of reliably predicting response times in real time of the single server shortest remaining processing time queue or the G/G/1/SRPT queue. Our proposed prediction methodology is to combine stochastic simulation with supervised learning machine learning algorithms. We consider several forms of state information like the job size or the number of other jobs in the system at the time of arrival. We hope that our methodology can be replicated for prediction purposes in other queueing systems.

1 INTRODUCTION

The Big Data revolution has transformed the way we think and process data on a massive scale. Data centers are widely used for big data analytics, which often involve data-parallel jobs, including query and web service. A large data center typically has thousands of servers, running jobs that process a massive amount of data daily. The generators of these jobs are ever-demanding users who have very little patience and even want information about how long their job will take. With many large players in the data market, it has never been more important to be able to provide customers with accurate estimates about their processing times. These estimates help customers decide which platform they will use since they are eager to get their jobs processed quickly. This fundamental problem of response time prediction for data centers motivates this work.

In this paper, we analyze the simplest queueing model to understand the effectiveness of prediction methods on the simplest of queues. More specifically, we analyze the single-server queue operating under the shortest remaining processing time (SRPT) service discipline. In this G/G/1/SRPT queue, jobs arrive to the queue according to a general arrival process. Each such job has associated with it a processing time, which is a random variable that represents the amount of time that the server must spend working on this job to complete its service. As we care about prediction in this paper, we do not make any assumptions about the service time distribution other than it is non-negative and has no mass at zero. We allow for even dependence between the arrival and service distributions.

In an SRPT queue, jobs are served one at a time such that the job with the shortest remaining processing time is served first. In particular, upon completing the service of a given job, the server then takes into service the job in system with the shortest remaining processing time. This is done with preemption so that when a job arrives with a processing time that is smaller than the remaining processing time of the

978-1-6654-3311-2/21/$31.00 ©2021 IEEE

job in service, the server places the job in service on hold and begins serving the job that just arrived. Processing is done in a non-idling fashion so that the server idles only when the system is empty. The success of SRPT is because of its many optimality properties, see for example Schrage (1968), Smith (1978), Schrage and Miller (1966), Schassberger (1990). In particular, it is the service discipline that minimizes queue length across all other service disciplines. However, there are many drawbacks to SRPT since it has a large memory requirement for implementation since remaining processing times of all jobs in the queue must be known. Moreover, SRPT is known to cause "starvation" of large size jobs, see for example Schreiber (1993), Wierman and Harchol-Balter (2003), Bansal and Harchol-Balter (2001), Harchol-Balter et al. (2003).

Traditionally, the SRPT policy has been studied mainly in the computer science literature. For example, Bansal and Harchol-Balter (2001) study fairness for SRPT and Wierman and Harchol-Balter (2003) provides a framework for comparing policies in the M/G/1 setting. More recently, work by Down et al. (2009b), Down et al. (2009a), Gromoll et al. (2011), Gromoll and Keutel (2012) study the SRPT queue using heavy traffic limit theory and derive either fluid limits or diffusion limits for characterizing the sample path behavior with dynamical systems or Brownian motion type models. Most of the work shows that the SRPT queue can be described by reflecting Brownian motion in the diffusion limit sense or by simple fluid models that can be used to construct approximate state-dependent response times of jobs entering the system. Finally, Nuyens and Zwart (2006), Wierman and Zwart (2012) use large deviations theory to analyze the SRPT queue to do optimal tail scheduling.

Despite all of the beautiful theory about the SRPT queue, there is no work that tries to develop a data-driven algorithm for predicting response times of the SRPT queue. This is an important problem because both large and small jobs would like to know when their job will be completed. This is also especially important as the world is using more cloud computing resources. Since SRPT is known for starvation of large jobs, it would be very valuable to know when larger jobs could expect to depart the queue in real time. Moreover, when limited information is available, it is an important question to know whether response times can be reliably predicted from basic queueing information. Although there is no work in the SRPT setting, there is one paper by Ward and Whitt (2000) that attempts to predict response times in the single server processor sharing queue and a few papers in the FCFS setting i.e. Whitt (1999b), Whitt (1999a), Ibrahim and Whitt (2009), Shah et al. (2019) and Palomo and Pender (2020). All of these papers are inspired by mathematics instead of being data-driven. It is our intent in this work to be data-driven and not analytical. Despite that, there are some papers that are similar in spirit to our work like Tanash et al. (2019), Cunha et al. (2017), Rodrigues et al. (2016), Pasdar et al. (2020) and Duc et al. (2019).

What makes the prediction problem interesting is that unlike the G/G/1/FIFO queue, SPRT response times are not necessarily known at the time of arrival. This is simply because the response time can depend on the size of jobs that arrive after a job, there is no simple recursion to calculate the response time at the time of arrival. This feature of SRPT queues is precisely what makes it interesting to study from a prediction point of view and is one of the main goals of this work. Thus, our goal is to provide a new framework for response time prediction in single server queues with the SRPT queueing discipline using simulation as a data generator and machine learning as a predictive tool. By combining simulation and machine learning, we are able to develop new ways of predicting response times in SRPT queues and give incoming job reliable estimates of departure. Finally, our data driven approach does not assume anything about the dependence of the arrival and service distributions. Much of the theoretical work assumes Poisson arrivals, however, recent literature shows that arrivals may not be Poisson, see for example Sriram and Whitt (1986), Daw and Pender (2018b), Koops et al. (2018), Daw and Pender (2018a), Saha et al. (2019), Daw et al. (2020).

1.1 Contributions of Our Work

By using stochastic simulation for the G/G/1/SRPT queue and leveraging machine learning methods, we provide new insights to the following questions:

- How well can we predict response times for the G/G/1/SRPT queue using machine learning?
- What information is most important for predicting the response times for the G/G/1/SRPT queue?
- What machine learning methods work best at predicting the response times of the G/G/1/SRPT queue?

1.2 Organization of the Paper

The remainder of the paper is organized as follows. Section 2 introduces the SRPT queue and gives a brief history of how it has been analyzed in the applied probability literature. In Section 3, we present our simulation and machine learning results for various machine learning methods. We also explain how various parameters of the queueing model affect the performance of predicting the response times of SRPT queues. Finally, a conclusion and some future directions of research are given in Section 4.

2 SIMULATING THE SRPT QUEUEING MODEL

Here in this section, we explain how to simulate the SRPT queueing model. First one should start with an array or vector of arrival and job sizes. The best way to simulate this queue is by constructing a heap that keeps track of the shortest remaining processing time along with the index of the job. If one does this then all of the current jobs in the queue are sorted appropriately by their remaining processing times. With this structure, then it reduces to observing that only two potential events can happen. The first event is an arrival of a job in which one adds this new job to the heap. If the job is not shorter than the job currently at the server, then the server continues to process the job in service. However, if the job size is smaller than the remaining processing time of the job currently in service, then the server switches to the new job and starts to process it. The second type of event is a departure and in this case the head of the heap is removed and the next element begins processing if there is one to be processed.

To the authors knowledge, there are no papers that discuss response time prediction for SRPT queues in real time. Most research explores the use of fluid and diffusion limits or Laplace transforms to construct an approximate expected response time, see for example Down et al. (2009b), Gromoll and Keutel (2012), and Gromoll et al. (2011). Moreover, the approximation might only depend on the number in the system and the job size and not other forms of information. One reason is that it is very difficult if not impossible to construct analytical estimates for the conditional mean response time given different forms of information other than the queue length or job size. To this end, we leverage machine learning to understand how well current machine learning methods can use this additional information in the context of response time prediction. Hence, we develop the following approach for generating response time predictions for the G/G/1/PS queue:

1. Construct a discrete-event simulation of the G/G/1/SRPT queue.
2. The discrete-event simulation is then used to generate the following quantities for each arrival, the job size, the queue length upon arrival, the response time, and the amount of work in the system upon arrival, relative queue length, and relative work upon arrival.
3. Partition the data generated from the discrete-event simulation into training and test sets.
4. Train machine learning methods on the training sets and develop a predictive model.
5. Use the predictive model generated from the training data and make predictions on the test data.
6. Evaluate different machine learning methods using the AMSE has a measure of quality.

3 MACHINE LEARNING METHODS FOR THE SRPT QUEUE

3.1 How the Data is Collected and Analyzed

This paper leverages simulation and machine learning together for prediction purposes. In this regard, we view the simulation part as essential because it is by which we generate the training data used by the

machine learning methods. Thus, the simulation step should be viewed as a data collection step. Unlike most data collection situations, via simulation, we can collect whatever data we want as long as it can be simulated. Before we go into more detail describing the various machine learning methods that we use to predict the response times, we will describe how we generate the training data via simulation. Moreover, we outline how we generate the numerical results that follow.

The experiments that follow were all trained on $10,000$ simulated response time observations and are implemented or tested on 10 independent sets for all methods and averaged. All numbers that are reported are based on the sample average mean squared error (AMSE) for the 10,000 samples i.e.

$$\text{Average MSE} = \frac{1}{n}\sum_{i=1}^{n}\left(R_i - \hat{R}_i\right)^2.$$

where

$$\hat{R}_i = f\left(X_i^{(1)}, X_i^{(2)}, X_i^{(3)}, X_i^{(4)}, X_i^{(5)}\right)$$

for some function $f(\cdot)$ and

- $X_i^{(1)}$ = size of the i^{th} job
- $X_i^{(2)}$ = queue length at time of arrival of the i^{th} job
- $X_i^{(3)}$ = amount of work in system at time of arrival of the i^{th} job
- $X_i^{(4)}$ = relative place in the queue at the time of arrival of the i^{th} job
- $X_i^{(5)}$ = amount of relative work at time of arrival of the i^{th} job
- R_i = the response time of the i^{th} job.

We should also mention that we performed all methods on the same data so the comparison between different machine learning methods is consistent and comparable. Moreover, we also simulated the queues using exponential and non-exponential distributions. For the exponential case, we assumed that the inter-arrival times occurred as exponential random variables with rate parameter λ and the job sizes were exponential with rate parameter μ. For the non-exponential case, we considered log-normal random variables for the inter-arrival times and job sizes given by $\text{LN}(\mu_a, \sigma_a)$ and $\text{LN}(\mu_s, \sigma_s)$ for the inter-arrivals and the job sizes respectively. For replication purposes, we specify what parameters we used for the exponential and non-exponential cases so that readers are able to replicate our results. Finally, all calculations were done using Python's sci-kit learn packages.

In our setting, we aim to predict the response time, R_i, of the i^{th} job by using information such as the job size $X_i^{(1)}$, the amount of work in the system at the time of arrival of the i^{th} job ($X_i^{(2)}$), and the queue length at the time of arrival of the i^{th} job $X_i^{(3)}$, the relative place in the queue at the time of arrival of the i^{th} job ($X_i^{(4)}$), and the amount of relative work at time of arrival of the i^{th} job ($X_i^{(5)}$). As $X_i^{(4)}$ and $X_i^{(5)}$ are not standard quantities, we mention that $X_i^{(4)}$ measures the place in the queue that the i^{th} job takes upon arrival when the job sizes are sorted appropriately and $X_i^{(5)}$ measures how much work is ahead of the i^{th} job takes upon arrival when the job sizes are sorted according to the SRPT discipline.

3.2 Linear Regression

In this section, we aim to understand how well linear regression will perform in predicting response times for the processor sharing queue. In the linear regression framework our goal is to find the coefficients $(\beta_0, \beta_1, \beta_2, \beta_3, \beta_4, \beta_5)$, in order to minimize the following loss function

$$\min_{\beta_0,\beta_1,\beta_2,\beta_3,\beta_4,\beta_5} \sum_{i=1}^{n}\left(R_i - \beta_0 - \beta_1 X_i^{(1)} - \beta_2 X_i^{(2)} - \beta_3 X_i^{(3)} - \beta_4 X_i^{(4)} - \beta_5 X_i^{(5)}\right)^2$$

in terms of the independent variables $\left(X_i^{(1)}, X_i^{(2)}, X_i^{(3)}, X_i^{(4)}, X_i^{(5)}\right)$. Linear regression is an attractive prediction method because it is quite simple and gives a natural interpretation of just weighting the available information upon arrival to construct a prediction. In fact, if one just receives the queue length as the information i.e $\beta_0 = \beta_1 = \beta_2 = \beta_4 = \beta_5 = 0$, then the linear regression can be interpreted as a statistical version of Little's law as it relates the queue length to the response time in a linear relationship. In fact, we will see this intuition that linear regression provides in the sequel of numerical experiments.

Table 1: Linear Regression Response Time Prediction M/M/1 Queue.

μ	1	1	1	1	10	10	10	10
λ	0.5	0.75	0.9	.99	5	7.5	9	9.9
Linear (AMSE)	1.32	13.72	131.79	8919.57	7.36e-3	.16	.85	224.72
β_0	-.62	-.52	-4.79	2.89	-.034	-.076	-.15	-1.23
β_1	1.59	2.75	5.64	20.20	1.48	2.84	5.23	23.76
β_2	.27	.53	0.97	1.02	.022	.05	.10	.26
β_3	-7.6e-3	-.050	-.16	-.084	-.024	-0.063	-.16	-.30
β_4	-.43	-1.67	-7.28	-187.57	-0.037	-.17	-.51	-24.41
β_5	1.86	3.83	11.12	214.33	1.74	3.71	8.07	297.40

Table 2: Linear Regression Response Time Prediction M/M/1 Queue.

μ	100	100	100	100	1000	1000	1000	1000
λ	50	75	90	99	500	750	900	990
Linear (AMSE)	1.18e-4	.014	1.24e-3	.96	9.39e-6	1.80e-5	5.05e-3	.022
β_0	-8.03e-3	.018	-.02	-.18	-2.11e-4	-1.43e-3	-3.53e-3	-5.41e-3
β_1	1.56	2.62	2.60	28.50	1.48	2.85	6.00	22.90
β_2	2.47e-3	4.52e-3	4.71e-3	0.018	2.71e-4	5.68e-3	9.55e-4	1.61e-3
β_3	-.017	-.028	-0.033	-.29	-.037	-0.054	-.10	-.21
β_4	-3.39e-3	-.019	-.019	-1.82	-4.93e-4	-2.05e-3	-.012	-.19
β_5	1.74	4.23	4.20	193.76	2.04	4.50	17.84	204.93

Table 3: Linear Regression Response Time Prediction G/G/1 Queue.

μ_a	1	1	1	1	10	10	10	10
σ_a	0.5	0.75	0.9	.99	5	7.5	9	9.9
μ_s	1	1	1	1	10	10	10	10
σ_s	0.5	0.75	0.9	.99	5	7.5	9	9.9
Linear (AMSE)	0.134	0.407	3.388	393.608	0.003	0.028	0.089	12.306
β_0	-0.112	-0.769	-1.867	-6.532	-0.030	-0.122	-0.210	-0.691
β_1	1.229	1.550	6.975	14.270	1.393	2.837	3.396	9.136
β_2	0.142	0.814	3.270	-0.185	0.028	0.008	0.024	0.075
β_3	-0.038	-0.163	0.447	-0.002	0.013	0.017	0.045	-0.093
β_4	-0.210	-3.749	-1.124	-43.23	-0.051	-0.061	-0.226	22.858
β_5	1.437	10.195	2.126	1.635	1.065	1.803	4.847	13.008

Table 4: Linear Regression Response Time Prediction G/G/1 Queue.

μ_a	100	100	100	100	1000	1000	1000	1000
σ_a	50	75	90	99	500	750	900	990
μ_s	100	100	100	100	1000	1000	1000	1000
σ_s	50	75	90	99	500	750	900	990
Linear (AMSE)	0.0002	0.0001	0.034	5.081	2.766	1.879	0.0002	0.305
β_0	-0.010	-0.004	-0.075	-0.201	-0.0002	-0.001	-0.004	-0.073
β_1	2.025	1.717	11.708	22.381	1.286	2.334	5.535	71.014
β_2	0.010	-0.0003	-0.004	-0.003	0.0001	0.0002	0.00005	-0.009
β_3	-0.110	0.098	0.111	0.013	0.065	0.046	0.017	1.560
β_4	-0.013	-0.008	-0.083	-1.008	-0.0002	-0.0004	-0.013	-0.791
β_5	2.691	2.928	-2.140	329.02	1.717	1.864	25.898	511.717

3.2.1 Discussion of Linear Regression Results

In this section, we summarize the performance of linear regression in the task of predicting response times in SRPT queues. In Tables 1 - 4, we observe that the performance of linear regression improves as we scale the arrival rate and service rate (large parameter setting) and we find that the performance degrades as we let λ approach μ or as we approach the heavy traffic regime. In all of the simulation experiments, we observe that the intercept i.e. β_0 is always negative. Moreover, as we increase the magnitude of the arrival and service rate parameters, the the value of β_0 approaches zero. We also observe that the coefficient β_1 is positive and is increasing as the offered load increases. This implies that as the job size increases, the response time increases significantly. The coefficient β_2, which measures the impact of the queue in the system upon arrival, is also positive, however, doesn't seem to impact the prediction as much as the job size. This is especially true as the arrival rate and service rate parameters are increased. The coefficient β_3 is pretty consistent in all parameter regimes and is also pretty small. The coefficient β_4 is mostly negative and seems to increase as the offered load increases. It also decreases as the rate parameters are increased as well. Finally, the coefficient β_5 is mostly positive, and seems to be quite helpful in prediciting the response times. Since it measures the relative work upon arrival, this is the minimum amount of time it would take for the job to finish, thus it is not surprising that that the coefficient is positive and takes values around in the interval [1,2] for most of the experiments.

3.3 Gaussian Processes

Table 5: Gaussian Processes Prediction M/M/1 Queue.

μ	1	1	1	1	10	10	10	10
λ	0.5	0.75	0.9	.99	5	7.5	9	9.9
Squared Exponential	7.02	18.43	158.35	1244.62	.058	.24	0.94	521.46
Matern 5/2	7.01	18.42	158.37	1244.62	0.058	0.23	0.94	521.46
Rational Quadratic	1.58	6.01	107.20	983.00	8.47e-3	0.086	0.47	511.41

Table 6: Gaussian Processes Prediction M/M/1 Queue.

μ	100	100	100	100	1000	1000	1000	1000
λ	50	75	90	99	500	750	900	990
Squared Exponential	.067	.013	.021	2.45	5.94e-3	8.34e-4	2.47e-4	.013
Matern 5/2	1.17e-3	4.41e-3	.021	2.44	3.54e-4	5.39e-5	2.09	.011
Rational Quadratic	2.18e-4	1.54e-3	.014	2.28	1.25e-6	1.21e-5	1.27e-4	.010

Table 7: Gaussian Processes Prediction G/G/1 Queue.

μ_a	1	1	1	1	10	10	10	10
σ_a	0.5	0.75	0.9	.99	5	7.5	9	9.9
μ_s	1	1	1	1	10	10	10	10
σ_s	0.5	0.75	0.9	.99	5	7.5	9	9.9
Squared Exponential	6.07	31.88	140.11	4582.71	.067	.29	2.15	1650.54
Matern 5/2	5.85	31.77	140.00	4582.70	.060	.28	2.15	1650.05
Rational Quadratic	1.19	16.35	99.01	3634.69	0.011	0.13	1.46	1629.85

Table 8: Gaussian Processes Prediction G/G/1 Queue.

μ_a	100	100	100	100	1000	1000	1000	1000
σ_a	50	75	90	99	500	750	900	990
μ_s	100	100	100	100	1000	1000	1000	1000
σ_s	50	75	90	99	500	750	900	990
Squared Exponential	.25	.11	.17	24.20	.15	1.22e-3	.50	.083
Matern 5/2	6.86e-4	2.39e-3	.17	23.85	8.13e-6	2.76e-5	3.89e-4	.081
Rational Quadratic	6.73e-5	6.75e-4	.16	22.10	1.43e-6	1.14e-5	3.48e-4	.10

3.3.1 Discussion of Gaussian Processes Results

Here we summarize the performance of Gaussian processes on predicting the response times of the SRPT queue. In Tables 5 - 8, we observe that as we increase the arrival rate and service rate, Gaussian process regression performs much better. This is to be expected and was also observed in other methods. Moreover, we also observe that the performance gets worse as λ approaches μ and we approach the heavy traffic regime. We compare three different kernels using the default settings (squared exponential, Matern 5/2, rational quadratic) and observe that the rational quadratic performed uniformly better than the other kernels. Finally, we do not observe any significant difference in the results between the M/M/1 queue and the G/G/1 queue with log-normal random variables.

3.4 Deep Neural Networks

Table 9: Deep Neural Network Prediction M/M/1 Queue.

μ	1	1	1	1	10	10	10	10
λ	0.5	0.75	0.9	.99	5	7.5	9	9.9
ReLU	1.07835	1.92975	94.384	4943.8563	0.00679	0.1329	0.62111	5.1323
Tanh	0.41415	4.18596	20.515	534.5432	0.00395	0.6777	0.29138	9.4126
Sigmoid	0.18006	21.65124	22.668	83.7669	0.01354	0.1506	0.35792	12.4945

Table 10: Deep Neural Network Prediction M/M/1 Queue.

μ	100	100	100	100	1000	1000	1000	1000
λ	50	75	90	99	500	750	900	990
ReLU	0.00012	0.00109	0.0215	0.6805	0.00000190	0.000024	0.00028	0.0121
Tanh	0.00036	0.00380	0.0086	0.0257	0.00023908	0.0005236	0.00013	0.03375
Sigmoid	0.00098	0.00137	0.0261	0.0027	0.00162247	0.001627	0.00240	0.0016

Table 11: Deep Neural Network Prediction G/G/1 Queue.

μ_a	1	1	1	1	10	10	10	10
σ_a	0.5	0.75	0.9	.99	5	7.5	9	9.9
μ_s	1	1	1	1	10	10	10	10
σ_s	0.5	0.75	0.9	.99	5	7.5	9	9.9
ReLU	0.59	5.17	3.59	594.38	0.017	0.042	0.28	0.54
Tanh	0.88	4.79	5.28	565.34	0.010	0.044	0.27	0.77
Sigmoid	0.43	1.79	4.23	584.34	0.046	0.078	0.41	0.74

Table 12: Deep Neural Network Prediction G/G/1 Queue.

μ_a	100	100	100	100	1000	1000	1000	1000
σ_a	50	75	90	99	500	750	900	990
μ_s	100	100	100	100	1000	1000	1000	1000
σ_s	50	75	90	99	500	750	900	990
ReLU	6.01e-4	1.74e-3	9.87e-3	.084	8.79e-4	4.14e-4	5.59e-4	.022
Tanh	3.40e-4	1.07e-3	.011	.087	1.23e-4	5.00e-4	1.61e-4	0.023
Sigmoid	9.28e-4	1.84e-3	0.010	0.087	1.61e-3	1.61e-3	1.70e-3	0.025

3.4.1 Discussion of Deep Neural Networks Results

Here we summarize the performance of deep neural networks on predicting the response times of the SRPT queue. In our deep neural network example we used 100 layers and 5 nodes for each experiment. Our results are given in the Tables 9 - 12 and we observe that the large parameter setting is easier to predict reliably. We also observe that the performance decreases significantly as we let λ approach μ. We also see that the performance is not significantly different for the M/M/1 queue vs. the G/G/1 queue. We compare three different activation functions (ReLU, Sigmoid, Tanh) and observe that ReLu very slightly outperforms Sigmoid and Tanh.

3.5 K-Nearest Neighbors

The K-nearest neighbors approach is well known in the machine learning community, however, we want to stress here to the readers that it has important significance in the context of prediction for queueing systems and other stochastic processes. The first important thing to note is that the KNN has an interpretation as a conditional expectation. In some sense, KNN is finding the situation that is most similar to the current situation and averages around those points. From a queueing perspective, it is trying to find situations that are the most similar to the job at hand. Thus, it allows us to use the data to compute an approximate conditional expectation given the job input characteristics. We will show in the sequel that this is a powerful prediction technique.

Table 13: K-Nearest Neighbor Prediction M/M/1 Queue.

μ	1	1	1	1	10	10	10	10
λ	.5	.75	.9	.99	5	7.5	9	9.9
k = 5	1.34	7.21	119.35	4212.19	7.34e-3	.22	.55	163.45
k = 10	1.28	6.94	110.14	4695.73	7.22e-3	.21	.49	179.79
k = 25	1.28	6.93	112.95	5107.14	7.28e-3	.22	.56	190.99

Table 14: K-Nearest Neighbor Prediction M/M/1 Queue.

μ	100	100	100	100	1000	1000	1000	1000
λ	50	75	90	99	500	750	900	990
k = 5	1.48e-4	8.48e-4	8.27e-3	.46	1.04e-6	1.62e-4	6.55e-4	.013
k = 10	1.39e-4	8.16e-4	7.99e-3	.60	9.61e-7	1.50e-5	5.95e-5	.013
k = 25	1.41e-4	8.45e-4	8.25e-3	.74	9.88e-7	1.54e-5	6.13e-5	.015

Table 15: K-Nearest Neighbor Response Time Prediction G/G/1 Queue.

μ_a	1	1	1	1	10	10	10	10
σ_a	0.5	0.75	0.9	.99	5	7.5	9	9.9
μ_s	1	1	1	1	10	10	10	10
σ_s	0.5	0.75	0.9	.99	5	7.5	9	9.9
k = 5	.83	10.06	75.54	1723.58	8.94e-3	.082	0.69	601.30
k = 10	.82	9.77	76.28	1885.97	8.53e-3	.082	0.75	772.98
k = 25	.89	10.42	80.88	2143.26	9.05e-3	.092	0.78	895.03

Table 16: K-Nearest Neighbor Response Time Prediction G/G/1 Queue.

μ_a	100	100	100	100	1000	1000	1000	1000
σ_a	50	75	90	99	500	750	900	990
μ_s	100	100	100	100	1000	1000	1000	1000
σ_s	50	75	90	99	500	750	900	990
k = 5	5.43e-5	5.54e-4	0.069	4.11	6.52e-7	8.29e-6	1.311e-4	0.018
k = 10	5.43e-5	5.13e-4	0.067	5.55	6.60e-7	8.21e-6	1.388e-4	0.021
k = 25	6.12e-5	5.23e-4	0.068	7.64	7.25e-7	8.84e-6	1.52e-4	0.023

3.5.1 Discussion of K-Nearest Neighbor Results

Here we describe the performance of K-nearest neighbors on predicting the response times of the processor sharing queue. In Tables 13 - 16, we observe that the performance of K-nearest neighbors improves as we increase the arrival rate and service rate. We also observe that the performance becomes worse as we let λ approach μ. In contrast to the other methods, KNN also appears to work well in the small parameter regime setting. This is mostly because, KNN has a conditional expectation interpretation and we are finding the closest representatives to our test points. KNN also outperforms the linear regression when the rates are large and the performance is comparable to Gaussian processes and deep neural networks in the large parameter regime. We compared three different values of K (5,10,25) and observed that the performance is similar, but $K = 5$ is the best of the values we chose.

3.6 Summary of Numerical Results

In this work, we have illustrated that combining stochastic simulation as a data generator for machine learning can help predict the response times of the SRPT queue. In summary, we observe that K-Nearest Neighbors works very well at predicting the response times in both the small parameter and large parameter regimes. It appears to be the most robust method and given that it can be viewed as a conditional expectation, this is not surprising. Thus, in all it seems that one can use most machine learning methods with confidence, however, we recommend KNN as it is very fast and is quite accurate in many parameter settings.

One important observation we made was that sometimes the AMSE can be misleading since the large values can be caused by one or two large mistakes in prediction. This seems to be the case especially for Gaussian processes and deep neural networks. Perhaps it might be important to develop other performance measures where if the majority of the errors are close to zero, then the total discrepancy is close to zero. This would help us understand the true performance of the method on most of the points and not just a small few.

4 CONCLUSION & FUTURE DIRECTIONS

In this work, we propose leveraging the power of stochastic simulation and machine learning for response time prediction for SRPT queues. We find that K-nearest neighbors is a robust method that performs well for small and large parameter settings and also performs well in light and heavy traffic. However, all of the machine learning methods work reasonably well on predicting the response times. Although not pictured here for space considerations, we find that even the machine learning methods with high AMSE's actually do well on a large percentage of the data. It turns that a small few of the test points do very poorly and skew the data. Thus, we can have high confidence that for a large majority of the data points, the prediction will work quite well.

We hope to continue to do research in this particular domain and one future direction would be to extend this analysis to multi-server SRPT queues and networks of SRPT queues since it is clear in many applications that one server and one queue is not enough, see for example Grosof et al. (2018), Mailach and Down (2017) and Pender and Phung-Duc (2016). In the multi-server setting, it would be great to predict response times while incorporating the impact of many servers. Finally, another area for future work is the area of SRPT queues with time varying rates. Non-stationary queues are more realistic than their stationary counterparts and it is important to explore the impact of non-stationarity on the machine learning predictions.

ACKNOWLEDGEMENTS

We would like to thank Cornell University for providing a stimulating environment to do this research as well as the ENGRD 2700 course where all three authors met.

REFERENCES

Bansal, N., and M. Harchol-Balter. 2001. "Analysis of SRPT Scheduling: Investigating Unfairness". In *Proceedings of the 2001 ACM SIGMETRICS International conference on Measurement and modeling of computer systems*, 279–290.

Cunha, R. L., E. R. Rodrigues, L. P. Tizzei, and M. A. Netto. 2017. "Job Placement Advisor Based on Turnaround Predictions for HPC Hybrid Clouds". *Future Generation Computer Systems* 67:35–46.

Daw, A., A. Castellanos, G. B. Yom-Tov, J. Pender, and L. Gruendlinger. 2020. "The Co-Production of Service: Modeling Service Times in Contact Centers Using Hawkes Processes". *arXiv preprint arXiv:2004.07861*.

Daw, A., and J. Pender. 2018a. "An ephemerally self-exciting point process". *arXiv preprint arXiv:1811.04282*.

Daw, A., and J. Pender. 2018b. "Queues Driven by Hawkes Processes". *Stochastic Systems* 8(3):192–229.

Down, D. G., H. C. Gromoll, and A. L. Puha. 2009a. "Fluid Limits for Shortest Remaining Processing Time Queues". *Mathematics of Operations Research* 34(4):880–911.

Down, D. G., H. C. Gromoll, and A. L. Puha. 2009b. "State-dependent Response Times via Fluid Limits in Shortest Remaining Processing Time Queues". *ACM Sigmetrics Performance Evaluation Review* 37(2):75–76.

Duc, T. L., R. G. Leiva, P. Casari, and P.-O. Östberg. 2019. "Machine Learning Methods for Reliable Resource Provisioning in Edge-cloud Computing: A Survey". *ACM Computing Surveys (CSUR)* 52(5):1–39.

Gromoll, H. C., and M. Keutel. 2012. "Invariance of Fluid Limits for the Shortest Remaining Processing Time and Shortest Job First Policies". *Queueing Systems* 70(2):145–164.

Gromoll, H. C., Ł. Kruk, and A. L. Puha. 2011. "Diffusion Limits for Shortest Remaining Processing Time Queues". *Stochastic Systems* 1(1):1–16.

Grosof, I., Z. Scully, and M. Harchol-Balter. 2018. "SRPT for Multiserver Systems". *Performance Evaluation* 127:154–175.

Harchol-Balter, M., B. Schroeder, N. Bansal, and M. Agrawal. 2003. "Size-based Scheduling to Improve Web Performance". *ACM Transactions on Computer Systems (TOCS)* 21(2):207–233.

Ibrahim, R., and W. Whitt. 2009. "Real-time delay estimation based on delay history". *Manufacturing & Service Operations Management* 11(3):397–415.

Koops, D. T., M. Saxena, O. J. Boxma, and M. Mandjes. 2018. "Infinite-server queues with Hawkes input". *Journal of Applied Probability* 55(3):920–943.

Mailach, R., and D. G. Down. 2017. "Scheduling Jobs with Estimation Errors for Multi-server Systems". In *2017 29th International Teletraffic Congress (ITC 29)*, Volume 1, 10–18. IEEE.

Nuyens, M., and B. Zwart. 2006. "A Large-Deviations Analysis of the GI/GI/1 SRPT Queue". *Queueing Systems* 54(2):85–97.

Palomo, S., and J. Pender. 2020. "Learning Lindley's Recursion". In *Proceedings of the 2020 Winter Simulation Conference*, edited by K. Bae, B. Feng, S. Kim, S. Lazarova-Molnar, Z. Zheng, T. Roeder, and R. Thiesing, 184–191. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Pasdar, A., Y. C. Lee, and K. Almi'ani. 2020. "Hybrid Scheduling for Scientific Workflows on Hybrid Clouds". *Computer Networks* 181:107438.

Pender, J., and T. Phung-Duc. 2016. "A Law of Large Numbers for M/M/c/Delayoff-setup Queues with Nonstationary Arrivals". In *International conference on analytical and stochastic modeling techniques and applications*, 253–268. Springer.

Rodrigues, E. R., R. L. Cunha, M. A. Netto, and M. Spriggs. 2016. "Helping HPC Users Specify Job Memory Requirements via Machine Learning". In *2016 Third International Workshop on HPC User Support Tools (HUST)*, 6–13. IEEE.

Saha, A., N. Ganguly, S. Chakraborty, and A. De. 2019. "Learning network traffic dynamics using temporal point process". In *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*, 1927–1935. IEEE.

Schassberger, R. 1990. "The Steady-State Appearance of the M/G/1 Queue Under the Discipline of Shortest Remaining Processing Time". *Advances in Applied Probability*:456–479.

Schrage, L. 1968. "Letter to the Editor—A Proof of the Optimality of the Shortest Remaining Processing Time Discipline". *Operations Research* 16(3):687–690.

Schrage, L. E., and L. W. Miller. 1966. "The Queue M/G/1 with the Shortest Remaining Processing Time Discipline". *Operations Research* 14(4):670–684.

Schreiber, F. 1993. "Properties and Applications of the Optimal Queueing Strategy SRPT. A Survey". *AEU. Archiv für Elektronik und Übertragungstechnik* 47(5-6):372–378.

Shah, A., A. Wikum, and J. Pender. 2019. "Using Simulation to Study the Last to Enter Service Delay Announcement in Multiserver Queues with Abandonment". In *Proceedings of the 2019 Winter Simulation Conference*, edited by N. Mustafee, K.-H. Bae, S. Lazarova-Molnar, M. Rabe, C. Szabo, P. Haas, and Y.-J. Son, 2595–2605. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Smith, D. R. 1978. "A New Proof of the Optimality of the Shortest Remaining Processing Time Discipline". *Operations Research* 26(1):197–199.

Sriram, K., and W. Whitt. 1986. "Characterizing Superposition Arrival Processes in Packet Multiplexers for Voice and Data". *IEEE journal on selected areas in communications* 4(6):833–846.

Tanash, M., B. Dunn, D. Andresen, W. Hsu, H. Yang, and A. Okanlawon. 2019. "Improving HPC System Performance by Predicting Job Resources via Supervised Machine Learning". In *Proceedings of the Practice and Experience in Advanced Research Computing on Rise of the Machines (learning)*, 1–8.

Ward, A., and W. Whitt. 2000. "Predicting Response Times in Processor-sharing Queues". *Analysis of Communication Networks: Call Centres, Traffic, and Performance* 28:1.

Whitt, W. 1999a. "Improving Service by Informing Customers About Anticipated Delays". *Management science* 45(2):192–207.

Whitt, W. 1999b. "Predicting queueing delays". *Management Science* 45(6):870–888.

Wierman, A., and M. Harchol-Balter. 2003. "Classifying Scheduling Policies with Respect to Unfairness in an M/GI/1". In *Proceedings of the 2003 ACM SIGMETRICS international conference on Measurement and modeling of computer systems*, 238–249.

Wierman, A., and B. Zwart. 2012. "Is Tail-optimal Scheduling Possible?". *Operations research* 60(5):1249–1257.

AUTHOR BIOGRAPHIES

JAMOL PENDER is an assistant professor in the School of Operations Research and Information Engineering (ORIE) at Cornell University. He earned his PhD in the Department of Operations Research and Financial Engineering (ORFE) at Princeton University. His research interests include queueing theory, stochastic simulation, dynamical systems and applied probability. His e-mail address is jjp274@cornell.edu. His website is https://blogs.cornell.edu/jamolpender/.

SUKRITI SUDHAKAR is an undergraduate student in Operations Research and Information Engineering at Cornell University. Her research interests are in queueing theory, machine learning and stochastic simulation. Her e-mail address is ss2742@cornell.edu.

EVA ZHANG is an undergraduate student in Operations Research and Information Engineering at Cornell University. Her research interests are in queueing theory, machine learning and stochastic simulation. Her e-mail address is yz544@cornell.edu.