Homework 2

Professor Lydia Y.Chen

CS4215: - Quantitative Performance Evaluation for Computing systems

September 21, 2020

Exercise 1. (6 Points)
For the Jackson network in Figure 1, assume that $0 < p < q < 1$ and assume that r is chosen so as to not overload the network. Answer the following questions:

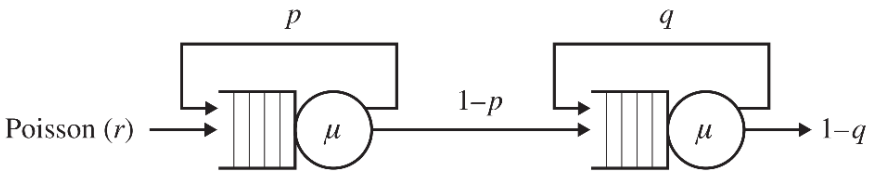

Figure 1: Figure for Exercise 1

1. Determine the mean response, $\boldsymbol{E}[T]$, as a function of r, μ, p and q.
2. If we interchange the order of the queues (i.e., the p and q are flipped), does $\boldsymbol{E}[T]$ increase, decrease, or stay the same?

Exercise 2. (10 Points)
In the *time.csv* file, the inter-arrival time at the first column and service time at the second are provided. They are derived from Pareto distribution. In the following, you can find the formulations of M/M/1 and G/G/1 systems, where the service rate is μ and arrival rate is λ. The mean time in the M/M/1 system and mean time in queue are:

$$E[T] = \frac{1}{\mu - \lambda}$$

$$E[T_Q] = \frac{\rho}{\mu - \lambda}$$

Also, an approximation for the mean waiting time in G/G/1 queue is:

$$E[W_Q] \approx (\frac{\rho}{1-\rho})(\frac{c_s^2 + c_a^2}{2})\tau$$

where τ is the mean service time (i.e. $\mu = \frac{1}{\tau}$ is the service rate), λ is the mean arrival rate, $\rho = \frac{\lambda}{\mu}$ is the utilization, c_a is the coefficient of variation for arrivals (that is the standard

deviation of arrival times divided by the mean arrival time) and c_s is the coefficient of variation for service times.
Your job is to simulate an G/G/1 queue by considering the input file information. Also, base on input file and provided formulation, calculate the waiting time for M/M/1 and G/G/1 to compare the result with simulation.

Figure 2: Figure for Exercise 7

Exercise 3. (10 Points)
Your system consists of a single CPU with finite buffer capacity. Jobs arrive according to a Poisson process with rate λ jobs/sec. The job sizes are Exponentially distributed with mean $\frac{1}{\mu}$ seconds. Jobs are serviced in FCFS order. Let $N-1$ denote the maximum number of jobs that your system can hold in the queue. Thus, including the job serving, there are a maximum of N jobs in the system at any one time (this is called an M/M/1/N queue). If a job arrives when there are already N jobs in the system, then the arriving job is rejected.
Your DARPA proposal requires that you reduce the loss probability in your system. To do this you could either ask for money to double the buffer size, or, alternatively, you could ask for money to double the speed of the CPU so that jobs get processed faster, thereby lowering the probability that there are N jobs in the system. Assuming both proposals have the same cost, which do you choose? (Asking for both makes you seem greedy.)
These are the specific questions you should answer:

1. Draw the CTMC and derive the limiting probabilities, then derive a closed-form expression for $\boldsymbol{E}$[Number in system].

2. Determine a closed-form expression for $\boldsymbol{E}$[T] for only those jobs that enter the system.

3. Suppose that $N = 5$, and $\rho = \frac{\lambda}{\mu} = 0.8$. Which would have the greater effect on lowering loss probability: doubling the buffer size or doubling the CPU speed?

Exercise 4. (5 Points)
Consider an M/M/k system, where the service rate at each server is $\mu = 0.75$. Fix system utilization at $\rho = 0.85$. Now increase the number of servers, k, as follows - 1, 2, 4, 8, 16, 32, 64 - adjusting the arrival rate λ, accordingly. For each number of servers, derive (i) the fraction of customers that are delayed and (ii) the expected waiting time for those customers who are delayed. We are just looking for numerical answers here. Feel free to write a math program to evaluate the needed summations. Explain the trend that you see.

Exercise 5. (10 Points)
Bianca observes that her database throughput drops when she runs too many transactions

concurrently (this is typically due to thrashing). She also observes that if she runs too few transactions concurrently, her database throughput drops as well (this is often due to insufficient parallelism). To capture these effects, Bianca models her time-sharing database system as an M/M/1/PS queue with load-dependent service rate, $\mu(n)$, where n denotes the number of concurrent transactions. The function $\mu(n)$ is shown in Figure 3.

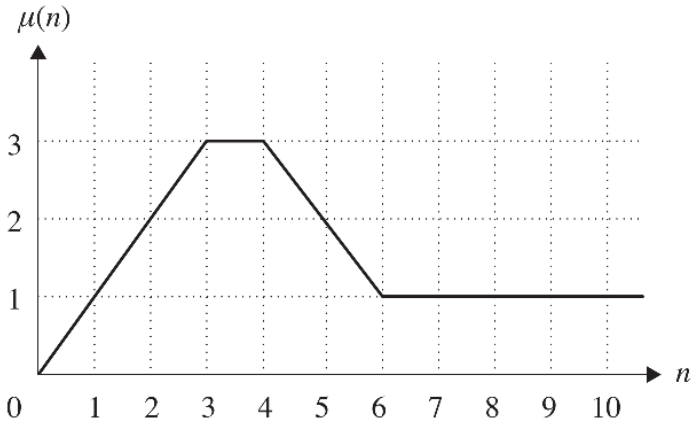

Figure 3: Figure for Exercise 3

1. Solve for the mean response time under Bianca's M/M/1/PS system. Assume arrival rate $\lambda = 0.9$. [Hint: Use a Markov chain.]

2. Bianca has a great idea: Rather than allow all transactions into the database as before, she decides to allow at most 4 transactions to run concurrently in the database, where all remaining transactions are held in a FCFS queue. Bianca's new queueing architecture is shown in Figure 4. Compute the mean response time for Bianca's new architecture, again assuming $\lambda = 0.9$, and Exponentially distributed service times with rates from Figure 4. What is the intuition behind Bianca's hybrid FCFS/PS architecture?

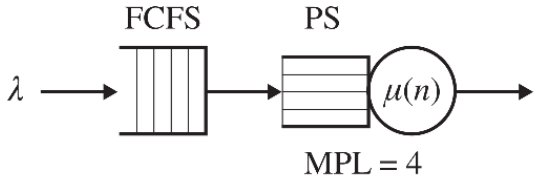

Figure 4: Processor-Sharing with limited multiprogramming level, MPL = 4.

Exercise 6. (**Bonus** - 10 Points)
Consider a router where, at each time step, the number of packets increases by 1 with probability 0.4 and decreases by 1 with probability 0.6. We are interested in the time

required for the router to empty. The Markov chain depicting the number of packets is shown in Figure 5. Let $T_{1,0}$ denote the time to get from state 1 to state 0. (a) Compute $\boldsymbol{E}[T_{1,0}]$. (b) Compute $\boldsymbol{Var}(T_{1,0})$. [Hint: The variance computation is a little tricky. Be careful not to lump together distinct random variables. Let $T_{i,j}$ denote the time to get from i to j. Importantly, note that $T_{2,0} \neq 2T_{1,0}$, since $T_{2,1}$ and $T_{1,0}$ need not be identical.]

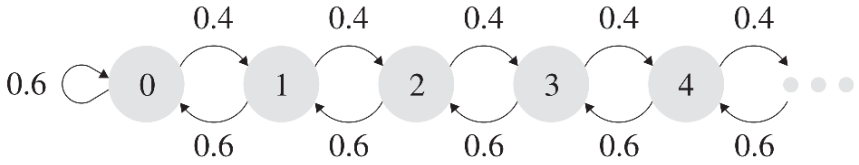

Figure 5: Number of packets at router.

Exercise 7. (30 Points)
Using the given dataset for IoT attacks prediction, the purpose of this assignment is to build several classifiers to predict the attacks. Therefore, the code is provided and you should fill in the blanks based on the following steps to build the classifiers and report the accuracy. Also provide the code.
The dataset addresses the lack of public botnet datasets, especially for the IoT. It suggests *real* traffic data, gathered from 9 commercial IoT devices. The dataset consists of 82500 data instances for train and 27500 instances for test. Each data has 115 attributes with total 11 categories. We ask you to use the dataset for multi-class classification: 1 class of *benign* associated with the label 0 and 10 classes of *attacks* with the labels from 1 to 10.

You can find the data at the following link:
`https://drive.google.com/drive/folders/1dQUZWC0OqiWpbJpQRMFKfpySBDSZoUdu?usp=sharing`

Pre-processing

1. Load the data for train and test and plot the class label population with pie plot. (For more analysis on parameter, if you want, you can concatenate the test and train data, shuffle them and split them randomly each time with the above mentioned sizes.)

2. If the dataset is not balanced with regard to the class labels, try to resample the data to achieve a balanced dataset.

3. Standardize the features.

4. (**Bonus**) Use Principal Component Analysis (PCA) to reduce the feature space dimension into two orthogonal components.

Train the Classifiers

Train the following classifiers by fining the suitable training parameters for each and report the test accuracy. Which classifier has the best performance?

1. Logistic Regression on the original (**Bonus**: and also PCA extracted features)
2. Support Vector Machine (SVM)
3. Decision Tree
4. Random Forest
5. Multi-Layer Perceptron (MLP)

Exercise 8. (14 points)
The aim of this exercise is to tune machine learning models' hyperparameters and to check if the change in the parameters has an statistical significance on the performance. Here assume that we have an noisy labeled dataset $U = \{(\mathbf{x}_j, \tilde{y}_j)\}$ and a weak and a strong labeler (the noise level is 30%). To train a highly accurate classifier, we want to identify the suspicious noisy samples and relabel them by the help of the labeler. Since relabeling is an expensive task, the goal is to find out which data to assign to each of the labelers to gain the highest accuracy within the labeling budget by training a deep learning model with the labeled data. The strong labeler $\mathcal{S}$ which can always give the true label, is $\mathfrak{c}$ times more expensive than the weak labeler $\mathcal{W}$ to label each data point. The problem with the weak labeler is that it sometimes fails to assign the true label. We have a limited budget B for labeling which should not be exceeded, i.e. $E_{\mathcal{W}}(t) + \mathfrak{c}E_{\mathcal{S}}(t) \leq B$, where $E_{\mathcal{W}}(t)$ and $E_{\mathcal{S}}(t)$ are the number of data samples that have been labeled by each labeler until the time t respectively. For each data point $\mathbf{x}_j$, we have $I(\mathbf{x}_j)$ that shows the informativeness of that data. Moreover, $L_V(t)$ shows the model's overall reliability at time t. We introduce a Q function that links all these three elements together: *sample informativeness*, *labeling cost*, and *model reliability*.

$$Q(\mathbf{x}_j) = \frac{L_V(t)}{(I(\mathbf{x}_j))^{\alpha}(\mathfrak{c}E_{\mathcal{S}}(t))^{\beta}}$$

If the Q function is higher than a threshold $\bar{Q}$, we need to query the strong labeler and otherwise the weak labeler.
By setting the cost $\mathfrak{c} = 5$ and the budget $B = 250$, find the optimal value of the threshold $\bar{Q}$, α, and β to achieve the higher accuracy. We provide the source code that calculates the value of $I(\mathbf{x}_j)$ and $L_V(t)$, and also the deep learning model. The source code also models the behaviour of the weak labeler. Therefore, all you need to do is to change the mentioned hyperparameters ($\bar{Q}$, α, β), consider 4 level for each, repeat each experiment 5 times, and test the effectiveness of them on the accuracy using three-way ANOVA test. You need to run the code with this command:

```
python main.py -s mnist -a 1 -b 1 -t 0.1
```

where a, b and t are corresponding values for α, β, and threshold $\bar{Q}$. For more information, watch the video uploaded in the course page explaining this exercise.

(Bonus:) This is an ongoing research. If interested, find a better way to balance between the cost of labeling and the accuracy. Can you identify another way to combine the components of the Q function?

Exercise 9. (15 points)
This is a review exercise and meant to showcase you how performance models can be applied to real systems. To complete this exercise, you need to strictly follow the review template available on the brightspace `https://brightspace.tudelft.nl/d2l/le/content/280410/viewContent/1919112/View`. You need to first select ONE of the following four papers which use different modeling approaches taught in the class for optimizing real systems. The details of this exercise and papers are provided on the `https://brightspace.tudelft.nl/d2l/le/content/280410/viewContent/1919112/View`. Make sure you check them before your start the exercise.

1. Model Driven Computational Sprinting
   http://web.cse.ohio-state.edu/ stewart.962/Papers/morris2018modeldriven.pdf

2. Optimus: An Efficient Dynamic Resource Scheduler for Deep Learning Clusters
   https://i.cs.hku.hk/ cwu/papers/yhpeng-eurosys18.pdf

3. Making Disk Failure Predictions SMARTer
   https://www.usenix.org/system/files/fast20-lu.pdf

4. CROSSBOW: Scaling Deep Learning with Small Batch Sizes on Multi-GPU Servers
   https://luomai.github.io/publication/2019-vldb-crossbow/2019-vldb-crossbow.pdf