AALTO UNIV. SCHOOL OF ELECTRICAL ENGINEERING — **Assignment 3**
COMNET Department — DL 9.12.2018
ELEC-E7460 Modeling and Simulation, fall 2018 — Lassila

Returning your solutions

Write a short report on the results. Return the report and your Mathematica files in a zip-file to the return folder on course web page in MyCourses (under Assignments section). The deadline for returning your solutions for this exercise is **Sunday 9.12.2018**.

Requirements for reporting:

- In the report, for each problem explain also briefly your solution.
- For each problem, provide the Mathematica file(s) that can be executed/evaluated, so that the assistant is able to easily verify that your solution works. It may be just one Mathematica file, where you have clearly documented each problem, or you can return each problem in its own Mathematica file.
- The assignment is graded 1, 3 or 5. Minimum requirement (for grade 1) is to complete problem 1.

All problems below are related to steady state properties. Remember then to implement some kind of initial transient control, e.g., by discarding a certain number of samples from the start. Note that we are not expecting a detailed analysis on the length of the transient in your results.

Problem 1

a) Your first task is to create a simulator for the dispatching problem as discussed in the lecture. New jobs arrive to the system according to a Poisson process with rate λ. There are two parallel queues with rates μ_1 and μ_2 having infinite buffers for the jobs. The service times in both queues are independent and exponentially distributed. Jobs in the queue are processed according to the FIFO discipline. As statistics, collect the delays of the jobs and in particular the overall mean delay of the jobs.

For the dispatcher function, you need to implement the following policies (as discussed in the lecture):

- Randomized load balancing policy: In this policy, an arriving job is routed to queue 1 with probability $\mu_1/(\mu_1 + \mu_2)$ and otherwise to queue 2.
- JSQ (Join-the-Shortest-Queue): Route the arrival to the queue with smallest number of jobs. In case of a tie, choose the queue randomly.
- W-JSQ: Route the arrival to the queue i with the smallest $x_i/\mu_i, i = \{1, 2\}$, where x_i is the number of jobs in queue i. Again, in case of a tie, choose the queue randomly.

Check that your simulator is working properly with the following parameters: $\lambda = 1.8, \mu_1 = 1, \mu_2 = 2$. The results should be (approximately):

- Randomized load balancing: overall mean delay 1.67 time units (exact result)
- JSQ: overall mean delay 1.21 time units (simulated result)
- WJSQ: overall mean delay 1.12 time units (simulated result)

Hint: Recall that from M/G/1 lectures you already have code for simulating the 2-class M/G/1 PS queue. You can use that as a starting point for your simulator.

b) Consider now the system with the following service rates $\mu_1 = 1$ and $\mu_2 = 4$. Thus, the system is stable if $\lambda < 5$. Simulate the system with the following load values $\rho = \{0.1, 0.3, 0.5, 0.6, 0.7, 0.8\}$. Recall that load $\rho = \lambda/(\mu_1 + \mu_2)$. Estimate the overall mean delay of the jobs for the different policies (randomized load balancing, JSQ, WJSQ) and the corresponding 95% confidence intervals.

Present your results in a table or a figure and compare them. You may also simulate more load points and plot the mean values in figure which may help the comparison. What do you observe and try to explain your observations? Which policy performs best/worst? How does the load affect?

Problem 2

a) Your next task is to also implement a simulator utilizing the idea of redundant requests, where jobs are replicated to multiple servers. Specifically, implement the so-called N-model as discussed on the lecture slides, where the system receives jobs at aggregate rate λ and an incoming job is placed in class R (i.e., it is replicated in queue 1 and 2) and with probability $(1 - p_R)$ it is placed in class A (i.e., it is not replicated and only placed in queue 2).

Check that your simulator works with the following parameters: $\lambda = 1.8, \mu_1 = 1, \mu_2 = 2, p_R = 0.9$. The exact solutions in this case are (see formulae in the slides): mean delay of non-replicated jobs $E[T_A] = 1.03$, mean delay of replicated jobs $E[T_R] = 0.83$ and overall mean delay $E[T] = 0.85$.

b) Similarly as in problem 1, consider now the redundant request N-model with the following service rates $\mu_1 = 1$ and $\mu_2 = 4$. Additionally, we utilize the fact that since the service times are independent and exponentially distributed, the optimal choice for replication is to always replicate, i.e., we set $p_R = 1$. Thus, the system is still stable if $\lambda < 5$. Simulate this N-model with the following load values $\rho = \{0.1, 0.3, 0.5, 0.6, 0.7, 0.8\}$. Recall that load $\rho = \lambda/(\mu_1 + \mu_2)$. Estimate the overall mean delay of the jobs and the corresponding 95% confidence intervals. Compare your results to what you observed in problem 1?

Problem 3

In this task, the objective is to study the impact of the variability of the service time distribution. In particular, we consider service times that are less variable than exponential to highlight the effect of this on the N-model performance. As discussed in the lecture material, smaller variability in the service times reduces the benefits from replication. On the other hand, in the N-model we can control by the parameter p_R the degree of replication to be used.

As a service time distribution with an easily controllable variance, we use the Erlang distribution with parameters k and μ, which represents a random variable that is distributed as the sum of k independent Exp(μ) distributed random variables, i.e., the mean value is k/μ. Thus, compared with the exponential distribution ($k = 1$), the variance reduces by a factor $1/k$ for the same mean. In Mathematica, an Erlang distributed random variable is represented by the command ErlangDistribution$[k, \mu]$.

a) Consider the following scenario: $\mu_1 = 1, \mu_2 = 4$ and $\lambda = 3$, thus the load $\rho = 0.6$. Let $k = 2$, i.e., the variance reduces by a factor $1/2$. On server 1, in order to have mean service time $1/\mu_1 = 1$, the Erlang distribution for the service times has parameters $k = 2$ and $\mu = k \cdot \mu_1 = 2$. On server 2, in order to have mean service time $1/\mu_2 = 1/4$, the Erlang distribution for the service times has parameters $k = 2$ and $\mu = k \cdot \mu_2 = 8$. Simulate the N-model for $p_R = \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0\}$ and estimate the mean delays. Also simulate the corresponding performance for the randomized load balancing, JSQ and WJSQ policies. Present your results, including the 95% confidence intervals either as a table or in a figure. What do you observe when comparing the results? What is the optimal value for p_R?

b) Now we reduce the variability even further by changing $k = 5$, i.e., variance is reduced by a factor $1/5$. On server 1, in order to have mean service time $1/\mu_1 = 1$, the Erlang distribution for the service times has parameters $k = 5$ and $\mu = k \cdot \mu_1 = 5$. On server 2, in order to have mean service time $1/\mu_2 = 1/4$, the Erlang distribution for the service times has parameters $k = 5$ and $\mu = k \cdot \mu_2 = 20$. Repeat the same simulation experiment as above in part a). What do you observe now? How did the decrease in variability affect the results?