Technical University of Munich, Chair of Communication Networks
Prof. Dr.-Ing. W. Kellerer, Alberto Martínez Alba, Mu He

Analysis, Modeling and Simulation of Communication Networks
SS 2018

Programming Assignment - Part 4

Submission Deadline: 29. May 2018

4 Correlation (95 Points)

From now on, your system should always be an M/M/1/S system, i.e., the inter-arrival time (IAT) and the service time should be exponentially distributed.

When analyzing simulation results, it is important to check for correlations. In this part, you will implement a version of the TIC, that is able to calculate the correlation coefficient. If you correlate two different arrays or counters, you speak of cross correlation. If you correlate an array with a shifted version of itself, you speak of auto correlation.

4.1 Implementation

You have to implement all functions in the classes *AutoCorrelationCounter* and *CrossCorrelationCounter*, both in file *counter.py*.

Task 4.1.1: CrossCorrelationCounter

10 Points

Implement all missing methods in this class. Note, that this counter should count two values at once, one value for every array. Another possible and efficient implementation would be incorporate three counters as inner-variables. The method *get_cor()* calculates the cross correlation between the two internal arrays. Use the method *get_cov()* (get covariance) for your implementation of the correlation. The degree of freedom should be N-1 when you calculate the variance.

Task 4.1.2: AutoCorrelationCounter

15 Points

Implement all missing methods in this class. Note the same things like in the previous exercise. In this case, the correlation coefficient is dependent on the “lag” between the one internal array. The degree of freedom should be N-1 when you calculate the variance. The “lag” parameter in auto-covariance/correlation stands for how much do you right shift the array of numbers. You should consider cyclic shift here. As an example, suppose the number array is [1, 2, 3, 4].

Then “lag=1” means the covariance/correlation of [1, 2, 3, 4], which is the original array, and [4, 1, 2, 3], which is the shifted array.

4.2 Verification

Do the coding for sections 4.2 and 4.3 in the file *part4_simstudy.py*.

Task 4.2.1: Auto Correlation

10 Points

Verify the correctness of your auto correlation counter by autocorrelating the following sequences:

- 1, -1, 1, -1, 1, -1, 1, -1,...
- 1, 1, -1, 1, 1, -1, 1, 1, -1,...

Hint: Think of a reasonable lag size. Choosing one lag size is not enough for a good verification.

4.3 Simulation Study III

Now, test your code for correctness with the provided test class *part4_tests.py*. Afterwards, have a look at the class *CounterCollection*. You see some correlation counters here, that count different correlations between arrays. The following correlation counters have already been implemented.

- Correlation between IAT and waiting time of a packet
- Correlation between IAT and serving time of a packet
- Correlation between IAT and system time (waiting time + serving time) of a packet
- Correlation between serving time and system time of a packet
- Auto-correlation of waiting time with lags ranging from 1 to 20

Task 4.3.1: Correlations in the DES

10 Points

Calculate the above correlations and auto correlations for a given system utilization of $\rho = 0.01, 0.5, 0.8, 0.95$. The queue size should be infinite (you can set it to 10000) and the simulation time should be 10000 seconds.

Task 4.3.2: The Visualization of Covariance/Correlation

10 Points

Reuse the scenario parameters (ρ, queue size and simulation time) of the previous task. Now instead of printing covariance/correlation values, we want to have a more direct visualization of the relationship between the following values.

- IAT and serving time of a packet
- serving time and system time of a packet

Please draw scatter plots with different ρs. In order to retrieve the values, you might have to get access to the inner variables of *CounterCollection*. You should have eight plots in total.

Task 4.3.3: Auto relation of Waiting Times

10 Points

Prepare two plots for the autocorrelation for lags ranging from 1 to 20 and a given system utilization of $\rho = 0.01, 0.5, 0.8, 0.95$. One plot is for N = 100 and the other one is for N = 10000, where N is the number of served packets. In other words, the packets that are dropped will not be considered. The queue size should be unlimited. Hint: For this, you have to implement the function *do_simulation_n_limit()* in class *Simulation*, such that the simulation stops after a total packet count of N. However, you can take most of the code from your regular simulation routine.

4.4 Analysis and General Questions

Answer the following questions separately and in full sentences. Explain your answers.

Task 4.4.1: Lag size

10 Points

What are reasonable lag sizes for the test sequences in task 4.2.1? What does the correlation coefficient say? Consider especially the extreme values!

Task 4.4.2: Correlation Times in the DES

10 Points

What conclusions can you draw from the correlation coefficients? Interpret all results of task 4.3.1. What can you say about whether packets have to wait or not? Explain!

Task 4.4.3: Visualization of Covariance/Correlation

5 Points

What information can you obtain from the scatter plots you draw in task 4.3.2? How does the information relate to the covariance/correlation values in task 4.3.1?

Task 4.4.4: Auto Correlation

5 Points

After checking the result of task 4.3.3, what conclusions could you drawn when you compare N = 100 and N = 10000? What is the problem with N = 100?

Total: 95 Points