Mathijs van der Linden

s2598442

MSc Technology and Operations Management  
Rijksuniversiteit Groningen

Supervisors: N. van Foreest & H. Balsters

“Contested Garment Consistent”   
A New Queueing Discipline

A Simulation Study

March 31, 2016

# Abstract

This research is towards developing a new queueing discipline in the category of fair queueing based on the “Contested Garment consistent” principles of fairness a term coined by Aumann (2002). In this research, this principle is translated into a queueing discipline (CGC), and its performance is compared to that of First-Come First-Served (FCFS) and Round-Robin (RR).

A simulation model compares the performance of the queueing disciplines. The performance of the queueing disciplines is measured using two Key Performance Indicators: average time per job spent in queue and the coefficient of variation (CV) in time per job. The model represents three job streams feeding one server. One job flow has a large number of jobs arriving per period while the other job stream have less arrivals per period. Several scenarios were tested to determine the difference between CGC and RR/FCFS.

The results from the simulation show that the CV as well as the average time jobs spent in queue on average is higher in CGC than it is in FCFS or RR. Furthermore, the idea that lies at the root of CGC is not translated to the results in such a way that it makes a viable queueing discipline. This is probably caused by the complexity and inconsistency of CGC.

Contents

[Abstract 1](#_Toc447223735)

[1. Introduction 4](#_Toc447223736)

[2. Research Framework 6](#_Toc447223737)

[2.1. Production Network 6](#_Toc447223738)

[2.2. Dynamic Queueing Disciplines 6](#_Toc447223739)

[2.2.1. First Come First Served 6](#_Toc447223740)

[2.2.2. Shortest Processing Time 7](#_Toc447223741)

[2.2.3. Earliest Due Date 7](#_Toc447223742)

[2.3. Static Queueing Disciplines 8](#_Toc447223743)

[2.3.1. Fair Queueing 8](#_Toc447223745)

[2.3.2. Round-robin 8](#_Toc447223746)

[2.4. CG consistent 9](#_Toc447223747)

[3. Methodology 10](#_Toc447223748)

[3.1. Formulate the Problem 11](#_Toc447223749)

[3.1.1. Hypotheses 11](#_Toc447223750)

[3.1.2. Simulation Setting 11](#_Toc447223751)

[3.2. Collect Information/Data and Construct Conceptual Model 12](#_Toc447223752)

[3.3. Validity of the Conceptual Model 14](#_Toc447223753)

[3.4. Program the Model 14](#_Toc447223755)

[3.4.1. FCFS 14](#_Toc447223756)

[3.4.2. RR 14](#_Toc447223757)

[3.4.3. CGC 14](#_Toc447223758)

[3.5. Programmed Model Validation 16](#_Toc447223759)

[3.6. Design, Conduct, and Analyse Simulation Experiments 17](#_Toc447223760)

[4. Results 18](#_Toc447223761)

[4.1. Performance Comparison 19](#_Toc447223762)

[4.2. Hypotheses Testing 22](#_Toc447223763)

[5. Discussion 23](#_Toc447223764)

[6. Conclusion 24](#_Toc447223765)

[Bibliography 25](#_Toc447223766)

[Appendix A. Model Testing 26](#_Toc447223767)

[Appendix B. Code 28](#_Toc447223768)

[Appendix C. Results 36](#_Toc447223769)

[Average Scenario Values 36](#_Toc447223770)

[Raw Results 37](#_Toc447223771)

[Figure 1. Production Network 6](#_Toc447223772)

[Figure 2. “A Seven-Step Approach for Conducting a Successful Simulation Study” by Law (2003) 10](#_Toc447223773)

[Figure 3. Simulation Setting 11](#_Toc447223774)

[Figure 4. Conceptual Model 13](#_Toc447223775)

[Figure 5. Total Queue Length per Queueing Discipline server 3, with smallest average stream 3 and seed 3 16](#_Toc447223776)

[Figure 6. Conceptual Model 18](#_Toc447223777)

[Figure 7. KPIs arrival [1,8,16] 19](#_Toc447223778)

[Figure 8. KPIs arrival [3,8,16] 20](#_Toc447223779)

[Figure 9. KPIs [5,8,16] 20](#_Toc447223780)

[Figure 10. KPIs arrivals [8,8,16] 20](#_Toc447223781)

[Figure 11. CV distribution 21](#_Toc447223782)

[Figure 12. Results of random selected period 26](#_Toc447223783)

[Figure 13. Results of a scenario 27](#_Toc447223784)

[Figure 14. Code of the simulation 35](#_Toc447223785)

[Table 1. Division of Estate to Creditors in “Ketubot” 93a as per the Mishna (Aumann, 2002) 9](#_Toc447223786)

[Table 2. Utilisation per server 12](#_Toc447223787)

[Table 3. Average Sum of avgD and CV 21](#_Toc447223788)

[Table 4. Average Sum of avgD and CV 24](#_Toc447223789)

[Table 5. Average results with arrivals [1,8,16] 36](#_Toc447223790)

[Table 6. Average results with arrivals [3,8,16] 36](#_Toc447223791)

[Table 7. Average results with arrivals [5,8,16] 36](#_Toc447223792)

[Table 8. Average results with arrivals [8,8,16] 37](#_Toc447223793)

[Table 9. Raw results from the simulation 39](#_Toc447223794)

# Introduction

The production industry is a highly competitive industry. Firms manage their production process closely to reduce cost and variability. Queueing theory is one of the aspects associated with system variability and is used for predicting queue lengths and waiting times of jobs through mathematical analysis (Sundarapandian & V., 2009). In queueing theory there are several queueing disciplines, sometimes called scheduling methods, consisting of an algorithm which determines the job order processing. The difference in queueing disciplines is reflected by the average waiting time of jobs and the variation of that waiting time. The choice for a queueing discipline hinges on the important characteristics of a situation (Hall, 1991).

First Come First Serve (FCFS) is a well-known example of a queueing discipline, however does not fit every situation. For example, imagine a rollercoaster ride where there are three queues. Queue A is for the regular crowd, queue B for the people with priority passes, and queue C for Very Important People (VIP). Here a FCFS discipline would not work since the people in queue B and C, that payed extra, have to wait for everyone person who arrived earlier than them. Fairness is an important characteristic in this situation, and FCFS does not fit this situation.

Another example of a queueing discipline, is strict priority. Recurring to the example, giving queue C strict priority will not please the people in queue A and B since they have to wait for the queue with the higher priority to empty. Hence, the fairness characteristic is not dealt with sufficiently. Nagle (1987), devised a fair queueing (FQ) discipline, which was used in computer networking. This resembled the method of Round-robin (RR). RR divides the capacity equally over all queues, however in the rollercoaster ride would this be deemed fair by the public? Hall (1991), states that for optimal effectiveness a fitting queueing discipline should be devised per situation. So is there another method of dividing the capacity of the rollercoaster ride which might be considered fairer?

In the Talmud[[1]](#footnote-1) a ‘fairness’ rule is described. This fairness rule is based on “equal division of the contested sum” p.4 (Aumann, 2002). The rule of the Talmud is coined by Aumann (2002) as the “Contested Garment consistent” (CG consistent), referring to the first example of this ruling in the Talmud. In modern law, proportional division is prominent in accordance to Aristotle’s Equity Principle, under which every dollar of debt is treated equal. Both principles can be considered fair but clearly take a different perspective on fairness. Aristotle’s Equity Principle is in essence similar to RR. The goal of this research is to adapt the CG consistent into a queueing discipline and compare the performance to RR and FCFS queueing discipline, because these are widely known and applied. Therefore, this research seeks to answer the following question:

*How can the CG consistent be translated into a queueing discipline and how does it perform in comparison to Round-robin and FCFS?*

To answer this question, these queueing disciplines are tested in a simulation which executes several scenarios. The performance of the queueing disciplines is measured by two KPIs which are: the coefficient of variation (CV) and average waiting time in queue per job. Followed by hypotheses to determine a statistically significant difference in performance. If there is a statistically significant difference and the performance of CG consistent queue discipline is satisfactory, the CG consistent queue discipline can be viewed as a new queueing discipline in the fair queuing category. Practically such a queueing discipline can be used in a multiple queue system feeding one server,

The next chapter consists of theory about queueing disciplines and sets the scope of the research. In chapter three the methodology is described, consisting of hypotheses, the simulation setting, the design for the simulation, and the validity of the model. Chapter four consists of the, and chapter five will discuss the results and give implications for future research. The last chapter concludes.

# Research Framework

In this chapter descriptions of several queueing disciplines are written. First an example of a production network is presented to visualise to the reader what kind of production network this research is suited for. This chapter will also include the functioning of the “Contested Garment consistent” which will be translated to a queueing discipline in chapter three. This research will not consider all queueing disciplines since they are numerous, especially in computer science. Only a small portion of queueing disciplines are described in this chapter, focussing on the well-known disciplines in manufacturing and the aforementioned Fair Queueing disciplines.

## Production Network

A queueing system combines several components, namely the arrival process, service process, and the queue (Hopp & Spearman, 2008). In Figure 1. an example of a production network is given. At one point, three streams link to one process. Consider three lines that all feed the same server. Job C (blue) has a high number of arrivals per hour. Job A (green) has a very low number of arrivals per hour in comparison, and B (red) an about average number of products per hour arriving. Suppose three jobs are waiting to be processed, the function of a queueing discipline is to determine in which order this will be done.

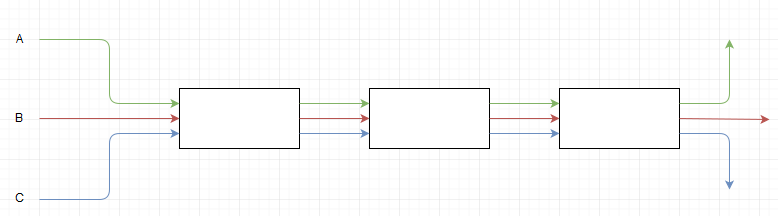


Figure . Production Network

Hall (1991), states queueing disciplines can be divided into two groups, static and dynamic disciplines. Ouelhadj & Petrovic (2009), have made further distinction among the dynamic queue disciplines, but for the purpose of this research this will be disregarded. Dynamic disciplines are updated constantly, meaning that with every job arrival a set of criteria are checked to determine which job is to be processed next. FCFS is an exmaple of a dynamic queueing system. Fair queueing is a static discipline, it determines over a period of time what jobs are to be processed in the next period of time and is not affected by job arrivals in that period.

## Dynamic Queueing Disciplines

Dynamic queueing disciplines are queueing disciplines which can be updated constantly. Every time a job arrives, the schedule will determine which job is to be processed next (Jackson, 1961). So priority is determined per job, but it is even possible to change queueing discipline with every job (Azarfar, Frigon, & Sansò, 2012). To follow is a number of dynamic queueing disciplines which are each briefly described.

### First Come First Served

In FCFS the job with the earliest arrival rate is selected when the server becomes available (Hall, 1991). This is probably the most well-known scheduling system and applied in many situations, for example in stores. Sometimes FCFS is described as a lack of a queueing discipline because this is a very ‘natural’ way of solving queues. FCFS minimizes waiting time variance for a single queue per server (Hall, 1991). In FCFS multiple queues can be treated as one, since the products are served based on their arrival time regardless what queue they are in (Bertrand, Wortmann, & Wijngaard, 1998). Therefore, FCFS relegates all flow control to the sources and thus when several queues feed the same server these job streams are not protected from each other.

As sketched in Figure 1. Production Network, there is a difference in arrival rate. Jobs A are unprotected from the frequent arrival of jobs C and B, and thus jobs A have to wait until all the jobs which arrived earlier are dealt with. Even though there might only be one job of A it might have to wait a long time before it is processed if a lot of other B and C jobs are waiting. Because every production system has variability (i.e. arrivals) in its sources, a queueing discipline should perform well regardless of an ill-behaving source (Demers, Keshav, & Shenker, 1989). Ill-behaving jobs claim a large portion of the capacity, which might not be desirable when other jobs are required for fulfilling orders, or should be considered higher priority.

### Shortest Processing Time

A quite popular queueing discipline used in manufacturing is the Shortest Process Time (SPT) (Hopp & Spearman, 2008) (Bertrand, Wortmann, & Wijngaard, 1998). The Shortest Process Time gives priority to the jobs which are processed in the shortest amount of time. In comparison to FCFS, SPT reduces the average cycle time. This is because the number of jobs in the queue will always be equal to or less than the number of jobs using any other queueing discipline with the same arrival and process times (Schrage, 1968). This effect increases when the utilization increases, when the variation of process time increases, or if the number of machines per workstation decreases (Bertrand, Wortmann, & Wijngaard, 1998).

### Earliest Due Date

Earliest Due Date (EDD) is a scheduling method with the characteristic of urgency (Goldberg, 1977). Such a discipline can be found in an emergency room. Patients who require immediate medical attention clearly benefit more from this discipline than FCFS. Yet the functioning of EDD is best when the due dates, for example measured in days, have little variance between them (Jackson, 1961). Determining a due date can be a hassle, unforeseen events can change the due date and complicate the use of this discipline (Ouelhadj & Petrovic, 2009).

## Static Queueing Disciplines

Static queueing disciplines determine what jobs are to be processed in a time period. To clarify, in Figure 1. Production Network three job flows feed one server. A static discipline determines what number of jobs each queue can release to the server. Yet which of these selected jobs is up first for processing is not pre-determined by such a discipline. Thus it is possible to assign priority within the selection of jobs to be processed. To follow is the description of several static queueing discipline.



### Fair Queueing

The main idea of Fair Queueing (FQ) is to assign each queue a fair fraction of the server capacity (Nagle, 1987), however what is deemed ‘fair’ is subjective. FQ has the advantage over FCFS that it considers the frequency of jobs sent (Demers, Keshav, & Shenker, 1989). FQs protect the job streams from each other, and prevents an arbitrarily large output of one job. In FQ disciplines the smallest stream has a lower cycle time and variance in comparison to FCFS (Demers, Keshav, & Shenker, 1989).

### Round-robin

Nagle (1987) introduced FQ in dividing capacity of switches in broadband networks. He divided the capacity equally amongst the inquiring parties. This version of FQ is very similar to Round-Robin (RR). RR is an algorithm commonly used in process and network scheduling. Instead of finishing jobs, RR schedules time slices in which the job is processed. It repeats this process for all periods until all jobs are finished. The job is halted and further processed in the next cycle if it is not finished in the assigned time slot (Arpaci-Dusseau & Arpaci-Dusseau, 2014). All the jobs are treated without priority (Nagle, 1987). This means the goal is to generate equal output of all jobs, unless overcapacity occurs in which the overcapacity is spread over the other queues. This is considered to be one fair way of handling the different jobs flows. The CG consistent is another fair way of dividing capacity and therefore would fall under the FQ discipline.

## CG consistent

The “Contested Garment consistent” (CGC) coined by Aumann p.6 (2002) refers to the first mentioned example of dividing debt in the Talmud. In this example two parties claim a garment, the first claims the whole garment and the second claims half of the garment. The first claimant receives ¾ and the second claimant receives ¼ of the garment. The logic behind this division is that there is only a dispute over half the garment, so each gets half of this part. The rest is for the largest claim.

The CG consistent does have several rules when dividing the debt which complicate the division:

* **Rule 1:** Equal amounts appointed to claimants if the value of the estate does not exceed half the sum of the claims, as long as this does not exceed half of their claim.
* **Rule 2:** Equal loss amongst claimants if the total estate value is greater than half the sum of the claim, creditors cannot lose more than half of their claim.

To demonstrate these rules the following example will be explained:

*A father leaves his estate to his three children after passing away. The value of an estate is to be shared amongst them, we will call them the creditors. The three creditors each have a different sized claim on the estate and depending on the value of the estate they get a different amount.* These amounts they would receive are shown in Table 1. Division of Estate to Creditors in “Ketubot” 93a as per the Mishna*.*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | | **Claim** | | |
| 100 | 200 | 300 |
| **Value Estate** | 100 | 33 ⅓ | 33 ⅓ | 33 ⅓ |
| 150 | 50 | 50 | 50 |
| 200 | 50 | 75 | 75 |
| 300 | 50 | 100 | 150 |
| 400 | 50 | 125 | 225 |
| 500 | 66 ⅔ | 166 ⅔ | 266 ⅔ |
| 600 | 100 | 200 | 300 |

Table . Division of Estate to Creditors in “Ketubot” 93a as per the Mishna (Aumann, 2002)

This division of the value is done in the following manners. First off is calculating half of the sum of the claims, which in this case is 300. So when the estate is valued at 150 rule 1 applies, and each claim gets the same amount (50). If the estate is valued at 200 the first claim receives 50, since the second part of the first rule is that the amount received cannot exceed half of their claim. The other claims receive what is left, after the first claimant receive their amount 200 – 50 = 150 is left. The second and third claim receive the same amount 150 / 2 = 75.

When the estate is valued at 500, thus over half the sum of the claims, the second rule applies. Meaning each claim loses the same amount of value, in this case 33 ⅓ (100 / 3). When the estate is valued at 400 there is a 200 deficit. The total of claims is 600 (600 – 400 = 200), therefore there is a 200 deficit which is spread equally over the three claims, so each claim would lose 66 ⅔ (200/3 = 66 ⅔). Yet this would mean that the smallest claim (100) would receive less than half of its claim (100 - 66 ⅔ = 33 ⅓). The second part of the second rule prevents this from happening. This part of the rule states that the maximum loss for a claim can only be half the claim, for the smallest claim this is 50, in this case. So with the smallest claim receiving 50, a 150 (200 – 50 = 150) deficit remains to be split amongst the other two claims. Thus, each of the other claims loses 75 of their claim (150/2 = 75).

# Methodology

The goal of this research is to compare the CGC to Round-Robin and First Come First Serve, which is represented by the following research question:

*How can the CG consistent be translated into a queueing discipline and how does it perform in comparison to RR and FCFS?*

This type of research is considered explanatory theory building research, since the goal is towards developing a new queueing discipline. In this research a statement about the viability of the CG consistent as queueing discipline will be made. To give such a statement, several disciplines are compared by means of simulation. In which is determined if there is a statistical significant difference in performance. Simulations have been used to test queuing disciplines by Demers, Keshav, & Shenker (1989) and Alomari & Menascé (2014) amongst others, showing simulation is a trial and tested method for this kind of topic.

The set-up of this research is based on Law’s (2003) model of simulation research, as shown below in Figure 2. “A Seven-Step Approach for Conducting a Successful Simulation Study” by Law (2003).

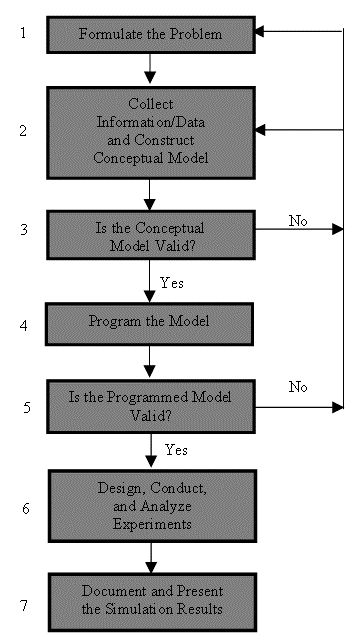


Figure . “A Seven-Step Approach for Conducting a Successful Simulation Study” by Law (2003)

## Formulate the Problem

*Formulate the Problem* is described in the introduction, and concerns the fair division of the capacity of a server over multiple queues. This research is towards developing a new queueing discipline, and comparing CGC to RR and FCFS. RR and FCFS each represent one of both categories of queueing disciplines, dynamic and static. FCFS and RR are chosen because of their wide and common application. Other queueing disciplines such as Shortest Process Time and Earliest Due Date are specified more towards a detailed situation. As this is an exploratory research with limited resources, a comparison to FCFS and RR seems sufficient in an initial research. The specific questions that will be answered in this research are represented in hypotheses.

### Hypotheses

Since CGC falls in the category of fair queueing, the same difference between other FQ’s and FCFS is expected. Thus, the expectation is that the average time jobs spent in the queue with the smallest average arrival rate is lower in CGC than FCFS. This is because the smallest queue is given a fraction of the capacity, and therefore is protected from more frequent arrivals from other sources.

*H1: CGC has a lower cycle time for the queue with the smallest job stream than the FCFS rule.*

The difference in performance between RR and CGC is expected to be in the spread of the CV of the three queues. In a system where there is a very small job stream, it is likely that the smallest queue in RR will have little to no variance. Every job that comes in is almost immediately processed, since this queue has a large proportion of the capacity in comparison to its arrivals, the last queue will probably show a high CV. CGC will probably distribute the variance in waiting time more equal amongst the queues than RR will.

*H2: The CV of CGC has less variance amongst the queues in comparison to the CV of the RR queues.*

The performance of queueing disciplines is translated to Key performance indicators (KPIs) which are:

* Average time spent in queue per job (avgD)
* The coefficient of variation (CV) of jobs per job type

### Simulation Setting

The scope in which this research is performed is depicted in Figure 3. Simulation Setting. The depicted model is an open queueing network were jobs arrive according to a Poisson distribution into the system. Several scenarios will be simulated in which the average of stream A changes per simulation scenario. The run time of the simulation will be set at 5000.

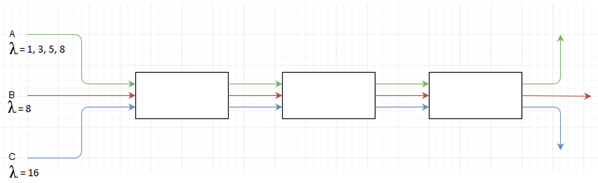


Figure . Simulation Setting

## Collect Information/Data and Construct Conceptual Model

The server capacity of the first server is set at the sum of the averages of all job streams plus three. Each subsequent server has one less capacity, thus the last server is the bottleneck of the line. This ensures the occurrence of queues for all servers. The utilisations will be as following:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 1 | 3 | 5 | 8 | Average |
| Server 1 | 89.29% | 90.00% | 90.63% | 91.43% | 90% |
| Server 2 | 92.59% | 93.10% | 93.55% | 94.12% | 93% |
| Server 3 | 96.15% | 96.43% | 96.67% | 96.97% | 97% |

Table . Utilisation per server

Variability is included in the arrivals of jobs which is according to a Poisson distribution, with mean λ, for each job flow (A, B, C). The average arrival for stream A is 1, 3, 5, or 8. For the simplicity of the model, failures and scrapping of jobs, and variability in capacity are not considered in the simulation.

Characteristics such as Work-in-Process (WIP) and Throughput (TH) are measured and used to calculate the Cycle Time (CT) using Little’s Law. Shown in the formula below.

The KPIs (average cycle time in queue and the CV) are calculated using the following formulas. Where average time jobs spent in queue is represented by , this variable is measured. *K* in these formulas represent the job number, determined by the job type and which queue it is residing in, *n* represents the period:

The standard deviation of waiting time (σ) per job is calculated as followed:

The coefficient of variation is calculated:

The hypothesis are tested by calculating the statistical significance. Statistical significance is tested using paired sample t-test with a 95% confidence interval. The following formula is used for the test:

Python 3.5 is used to simulate the performance of the queueing disciplines, this is a discrete event simulation tool. Discrete event simulation is chosen because RR and CGC do not require a queueing discipline, such as FCFS, to determine which of the jobs is handled first from the chosen jobs. All the previous information is visually represented in Figure 4. Conceptual Model.

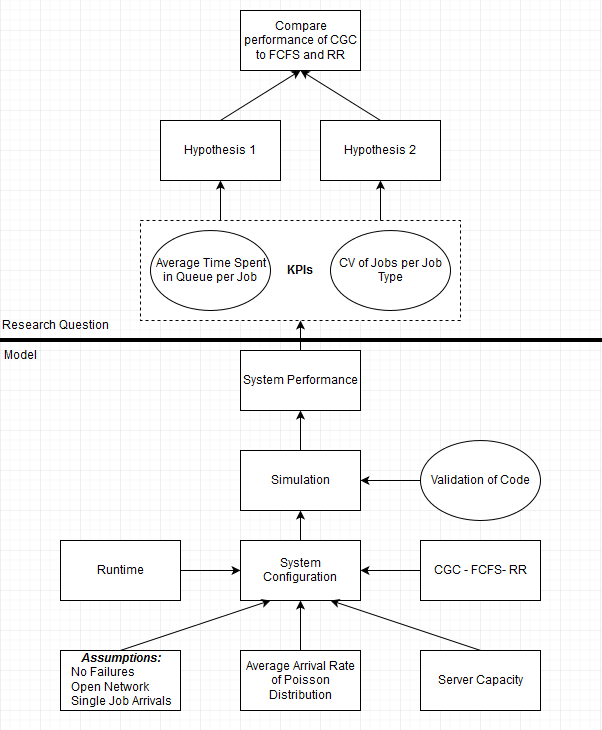


Figure . Conceptual Model

## Validity of the Conceptual Model

The conceptual model has been validated by the supervisor of this thesis.



## Program the Model

This chapter contains a mathematical or literal representation of how each queueing discipline is modelled into code and the translation of CGC into a queueing discipline. The simulation model is designed so that jobs are stored in arrays and each number in this array represents one job. The number represent the number of periods the jobs is in the system. Partial jobs cannot be processed, so in RR all the jobs are completely processed. There is a one period warm-up were no jobs depart the system.

### FCFS

In FCFS the three incoming streams are combined into one queue[[2]](#footnote-4). The job with the highest time in queue is picked to be processed. Since this concerns a discrete event simulation, several jobs will arrive at the same time. In such a case, all jobs with the highest waiting time are dealt with in a Round-Robin manner, but not leaving jobs partially finished. Then the queue for the next period can be determined as follows for all queueing disciplines. Let *Q* denote the number of jobs in the queue (*i*) at period *n*, let *A* denote the number of jobs arriving at the end of the period and *D* the number of jobs departing during the period. Then number of jobs in the queue is determined by:

The departures are determined by the amount of jobs that are in the queue or the capacity:

### RR

In Round-Robin the capacity is split equally over the three queues. Let *C* denote the server capacity and *I* (3) the total number of queues, then the appointed capacity (Ri) per queue *i*, at the start of the period is:

Meaning that the departures can be determined as:

The departure formula also applies to CGC. When the jobs in queue are less than the assigned capacity (Qn,I < Ri) the overcapacity of this queue is divided over the other queues so that no capacity is lost. When the division of capacity results in a number with decimals, the remainder is subtracted from all the number and this capacity is added to the queue with the largest number of jobs at that moment.

### CGC

Translating the CGC to a queueing discipline would mean the following. In CGC the length of the queue determines the size of the claim. The capacity of the server is what the claims dispute over. In Table 1. Division of Estate to Creditors in “Ketubot” 93a as per the Mishna (Aumann, 2002), the capacity can be viewed as the ‘estate value’. The amount of capacity given to a queue is done by the CGC.

For CGC first “half the sum of the claims” has to be determined to determine which ofH the two underlying rules applies. Half the sum of the claims (*yn*) is:

The server capacity which is appointed by CGC (Rn,i) is:

Which is subjected to a constraint depending on if “half the sum of the claims” exceeds the server capacity:

The first rule that underlie the CG consistent is represented in the following formula:

And the second rule:

## Programmed Model Validation

The validity of model is tested using the suggestions by Sargent (2003) and Law (2003). I will review the most important ones here, but an extensive list of the tests and their results can be found in Appendix At.

Predictive validity concerns the behaviour of the simulation in comparison to what is predicted when a change to the system is made. When increasing the number of queues, RR would still divide the capacity evenly which is the case in the simulation. The division of CGC seems legitimate as well with four queues. Furthermore in RR, the smallest job stream should have the lowest variability since this queue is very small in proportion to the assigned capacity and in comparison to the other queues. Which is also the case.

Degenerate tests are performed, which tests the degeneracy of the model’s behaviour. Setting the capacity closer to the average of arrivals shows that the average queue length increases. Setting the capacity equal to the average arrivals shows a complete flooding of the queues. Increasing the average arrivals results in a higher average number of jobs in queue.

Event validity concerns the occurrence in events and comparing these to real systems. Changing the average arrival rate affects the average number of jobs in queue. No negative number of jobs arrive, and empty queues do not have departures. Also the number of departures and the jobs left in queue are checked across all queueing disciplines if they match, this concept is a prerequisite for a proper functioning queueing system.

Extreme condition tests have been performed. This consisted of increasing the capacity of the server from very low to very high, in both situations the model performed as would be expected. When increasing the arrivals of the largest stream to an extreme the simulation acts accordingly. Traces are done through the model to determine if the modelled queueing discipline acts accordingly. For FCFS there should be no large difference between the waiting times of jobs, this has been verified manually by looking at approximately 25 random periods. The other queueing disciplines were also checked in similar fashion.

Internal validity is ensured by using a limited amount of variability. The variability in the arrival process helps representing a realistic situation. Variability in other parts of the model is left out because of limited resources and the increasing of complexity. Further variability could also cloud the results.

After running the simulation, some operational graphics are collected. Operational graphics can give insight into the correctness of the model. Figure 5. Total Queue Length per Queueing Discipline server 3, with smallest average stream 3 and seed 3 is a graph of the queue length. The total length of all queues should, and are equal.

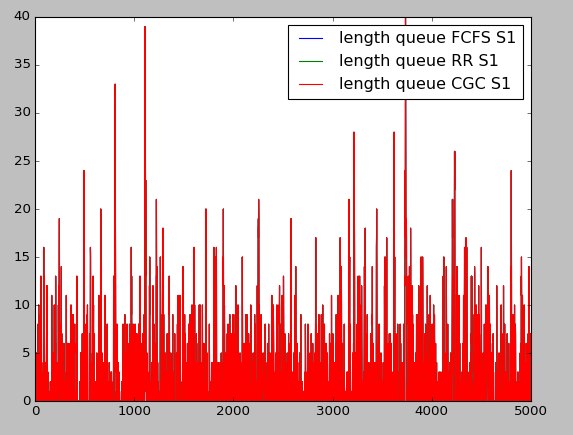


Figure . Total Queue Length per Queueing Discipline server 3, with smallest average stream 3 and seed 3

## Design, Conduct, and Analyse Simulation Experiments

The simulation consist of several scenarios, the difference between these scenarios are represented by the change of the average number of arrivals of the smallest stream. A scenario runs for 5000 periods to create a steady state in the system, which gives sufficient data to compare the KPIs. A scenario is run multiple times with different random seed values, which is 3, 5, and 7. This gives enough data to make accurate statements about CGC as a queueing discipline. Document and Present the Simulation Results is done in the next chapter.

Waarom doe je de test

Komt goed

# Results

The model described in the previous chapter was used to determine the KPIs of CGC, RR, and FCFS. In Figure 6. Conceptual Model, the relationship between the model and the research questions is shown and how the research was designed to compare the aforementioned performance.

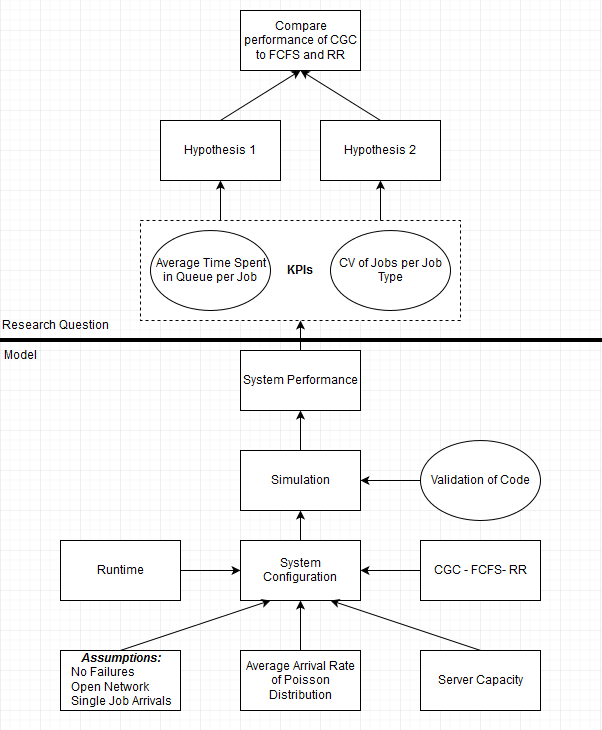


Figure . Conceptual Model

The results shown here are from the simulation. The validation of the code has proven that the simulation works accordingly. The code which is used for simulating the scenario can be found in Appendix B. Code.

## Performance Comparison

Queueing disciplines affect the average time jobs spent in the queue and the variation. To determine the performance the following KPIs were chosen:

* Average time spent in queue per job (avgD)
* The coefficient of variation (CV) of jobs per job type

In the following figure the KPIs are shown together, the difference between the figures is in the scenarios i.e. the average arrivals of the smallest stream[[3]](#footnote-5). In these figure, the queue is represented by a letter (a,b,c) and the KPIs by avgD and CV. AvgD meaning the average time spent in queue per job, and CV is the coefficient of variation (CV) of jobs per job type. On the x-axis the type of queueing discipline is shown and the number of the server is behind it (1,2,3). The utilisation of each server can be found in chapter 2.3. Table 2. Utilisation per server.

These results show that RR and FCFS operate according to expectation in comparison to each other. RR has a lower CV for first two queues and a higher CV for the third queue. RR protects the smaller streams from the largest stream, which FCFS does not do. Thus, these results are as expected in accordance with theory and confirm the validity of the simulation design.

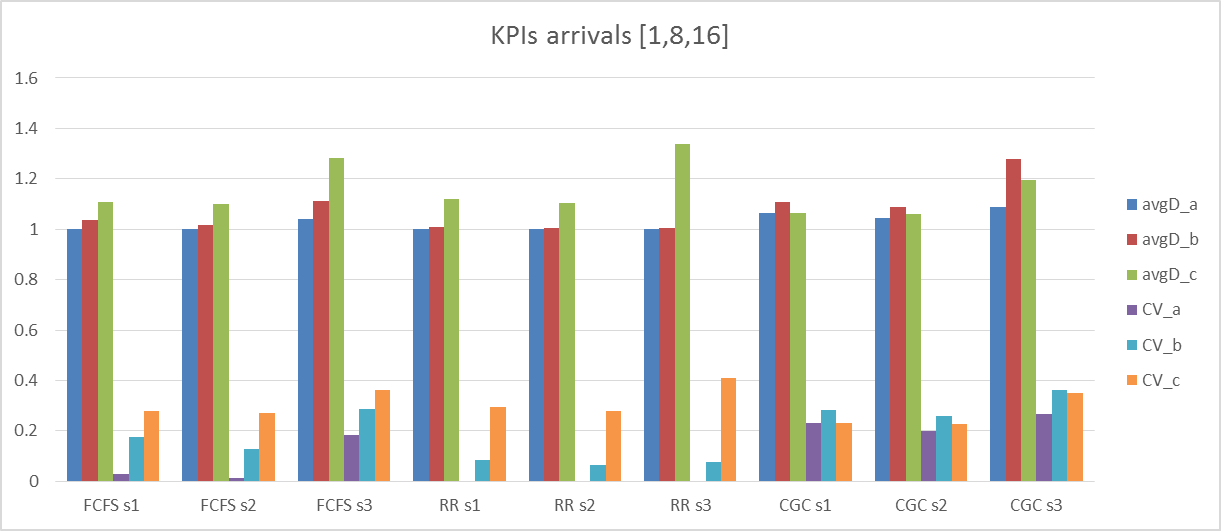


Figure . KPIs arrival [1,8,16]

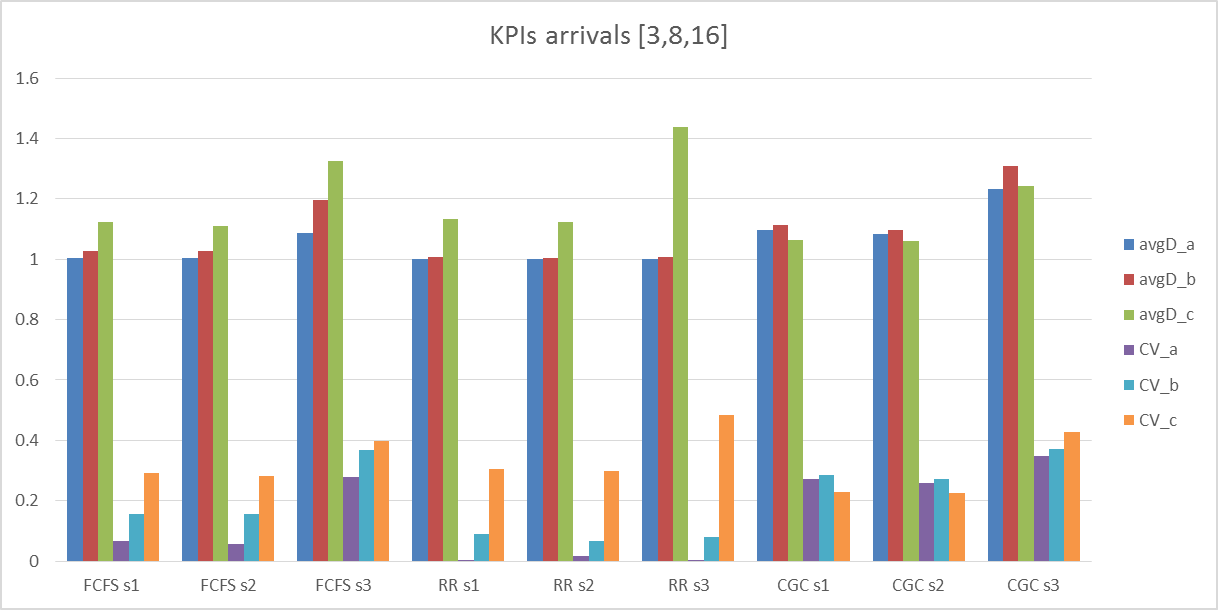


Figure . KPIs arrival [3,8,16]

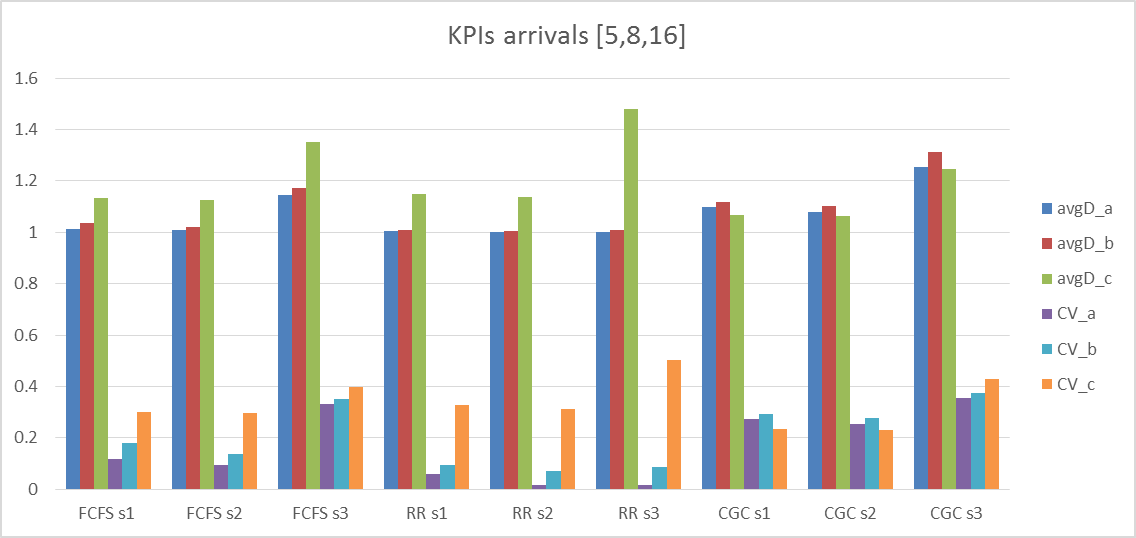


Figure . KPIs [5,8,16]

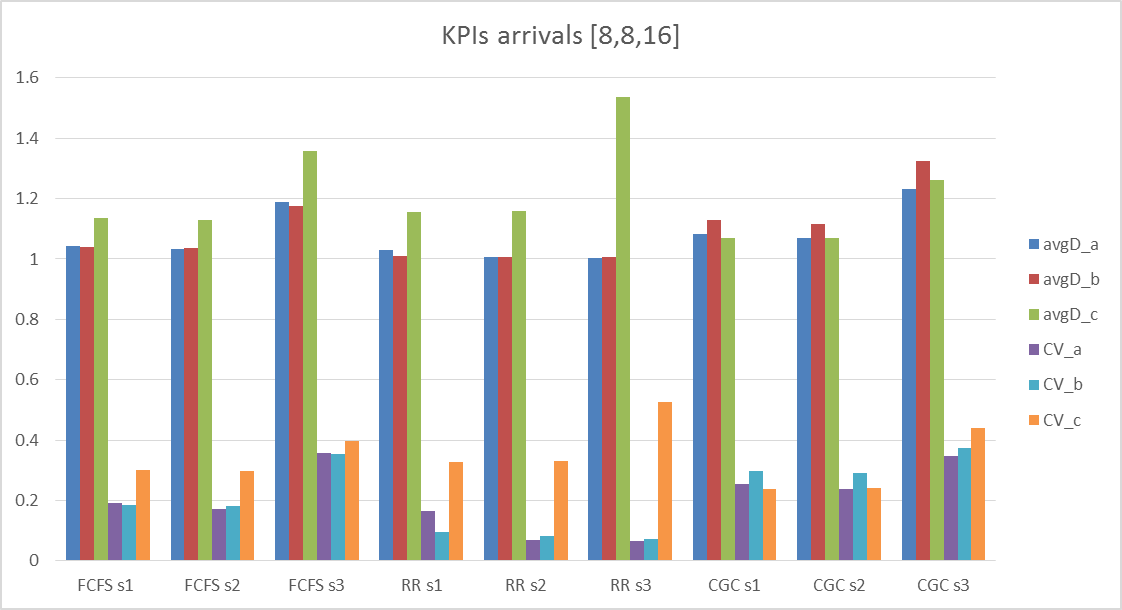


Figure . KPIs arrivals [8,8,16]

As expected the average time spent in queue per job in CGC looks to be fairly equal, however unexpectedly the middle queue has a larger average waiting time per job than the first and third queue, at server 1 and 2. The idea of CGC was to protect the smaller stream(s) from the other job stream and not ‘dump’ the CV of largest stream, as RR does. In CGC, the CV of the first two queues is higher than that of the other queueing disciplines in all scenarios. Which suggest that the performance of CGC is worse than the other queueing disciplines.

The sum of avgD and CV of CGC is higher than that of RR and FCFS in all scenarios as shown in

|  |  |  |  |
| --- | --- | --- | --- |
|  | FCFS | RR | CGC |
| avgD | 9.935823 | 9.749073 | 10.24744 |
| CV | 2.107633 | 1.441481 | 2.635491 |

Table 3. Average Sum of avgD and CV. Thus, CGC has an higher average waiting time for jobs and there is more variation in waiting time than there is in FCFS and RR.

|  |  |  |  |
| --- | --- | --- | --- |
|  | FCFS | RR | CGC |
| avgD | 9.935823 | 9.749073 | 10.24744 |
| CV | 2.107633 | 1.441481 | 2.635491 |

Table . Average Sum of avgD and CV

It was expected that the division of CV in CGC was different from that in RR and FCFS. The division of CV amongst the queues is shown in Figure 11. CV distribution. This figure shows that in most cases the CV of queue three in CGC is the lowest, but the CV of queues 1 and 2 is much higher. In server 3, which has an approximate utilisation of 97%, the performance of CGC and FCFS are quite similar however CGC still performs worse than FCFS in this case.

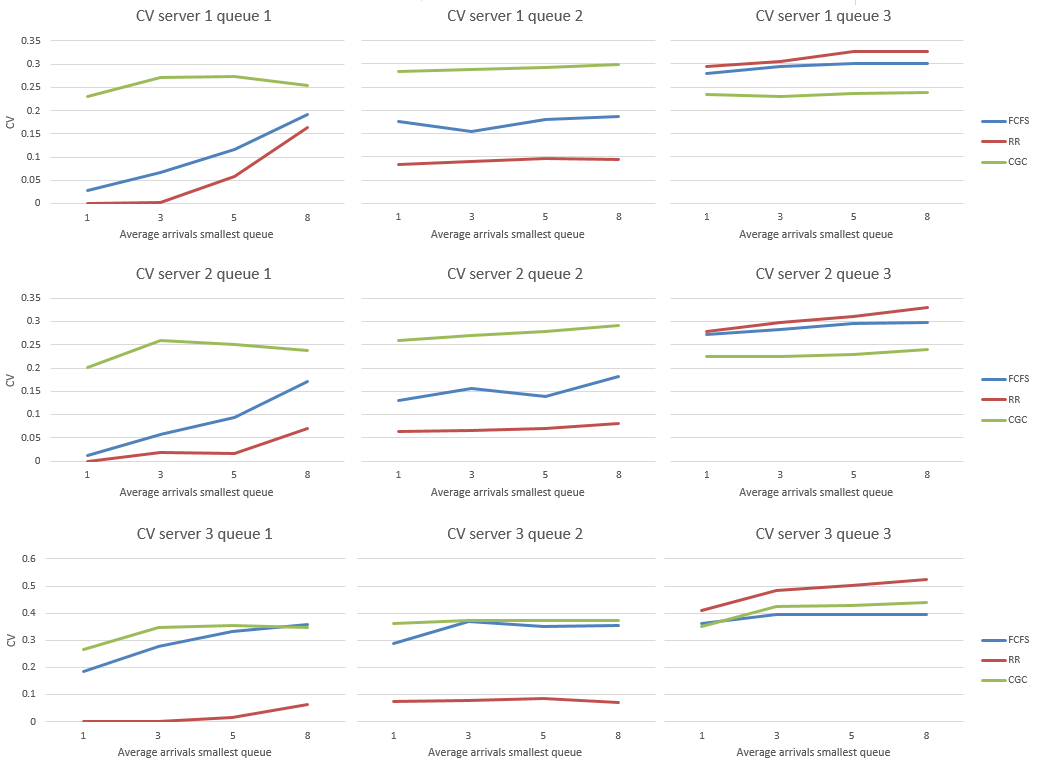


Figure . CV distribution

## Hypotheses Testing

## Program the Model

This chapter contains a mathematical or literal representation of how each queueing discipline is modelled into code and the translation of CGC into a queueing discipline. The simulation model is designed so that jobs are stored in arrays and each number in this array represents one job. The number represent the number of periods the jobs is in the system. Partial jobs cannot be processed, so in RR all the jobs are completely processed. There is a one period warm-up were no jobs depart the system.

### FCFS

In FCFS the three incoming streams are combined into one queue. The job with the highest time in queue is picked to be processed. Since this concerns a discrete event simulation, several jobs will arrive at the same time. In such a case, all jobs with the highest waiting time are dealt with in a Round-Robin manner, but not leaving jobs partially finished. Then the queue for the next period can be determined as follows for all queueing disciplines. Let *Q* denote the number of jobs in the queue (*i*) at period *n*, let *A* denote the number of jobs arriving at the end of the period and *D* the number of jobs departing during the period. Then number of jobs in the queue is determined by:

The departures are determined by the amount of jobs that are in the queue or the capacity:

### RR

In Round-Robin the capacity is split equally over the three queues. Let *C* denote the server capacity and *I* (3) the total number of queues, then the appointed capacity (Ri) per queue *i*, at the start of the period is:

Meaning that the departures can be determined as:

The departure formula also applies to CGC. When the jobs in queue are less than the assigned capacity (Qn,I < Ri) the overcapacity of this queue is divided over the other queues so that no capacity is lost. When the division of capacity results in a number with decimals, the rest value is subtracted from all the number and this capacity is added to the queue with the largest number of jobs at that moment.

### CGC

Translating the CG consistent to a queueing discipline would mean the following. In CGC the length of the queue determines the size of the claim. The capacity of the server is what the claims dispute over. In Table 1. Division of Estate to Creditors in “Ketubot” 93a as per the Mishna (Aumann, 2002), the capacity can be viewed as the ‘estate value’. The amount of capacity given to a queue is done by the CG consistent.

For CGC first “half the sum of the claims” has to be determined to determine which ofH the two underlying rules applies. Half the sum of the claims (*yn*) is:

The server capacity which is appointed by CGC (Rn,i) is:

Which is subjected to a constraint depending if “half the sum of the claims” exceeds the server capacity:

The first rule that underlie the CG consistent is represented in the following formula:

And the second rule:

, several hypothesis were posited which will be tested using the paired sample t-test:

The first hypothesis is:

*H1: CGC has a lower cycle time for the queue with the smallest job stream than the FCFS rule.*

Comparing the CGC to FCFS, results from the smallest queue gives a t-value of 7.62 which is associated with a p-value of 0.0001. As pre-determined, the confidence interval of 95% was chosen which gives an alpha of 0.05. Since p-value is smaller than alpha, there is a statistical significant difference in this KPI. However, the mean of the FCFS is lower than that of CGC and thus the cycle time of the smallest queue is lower in FCFS than in CGC. This means this hypothesis is rejected.

f-test

The second hypothesis is:

*H2: The CV of CGC has less variance amongst the queues in comparison to the CV of the RR queues.*

Comparing the variance of CV in RR and CGC results in a t-value of 292.94 which is associated with a p-value of 0.0001. As pre-determined, the confidence interval of 95% was chosen which gives an alpha of 0.05. Since p-value is smaller than alpha, there is a statistical significant difference and thus the hypothesis cannot be rejected.

In the next chapter the results are discussed in further depth.

# Discussion

The goal of this research was to research a possible new queueing discipline called CGC and how it would perform in comparison to FCFS and RR in a simulation model. The results are quite unexpected, the performance of CGC is far worse in every scenario than that of FCFS or RR. The jobs spent a higher average of time in queue and the coefficient of variation is higher as well. The idea behind CGC is the fairness of distributing the CV and average waiting time. Although the CV is distributed more evenly, compared to RR, it is also a lot higher than in RR or FCFS. CGC gives no indication that it is protecting the smaller streams, which is the main aspect as to why Fair Queueing was developed. These results suggest that CGC as a queueing discipline is inviable.

The inconsistency of CGC can be the reason for the bad performance. RR is a very straightforward method of selecting job, and only adjusts the division amongst the queues if the assigned capacity exceeds the number of jobs in that queue. CGC is a much more complex rule than RR. Meaning, there are a lot of aspects in determining the distribution of the capacity. CGC has two rules at its base, in certain periods in the simulation it switches between the rules and thus its method of assigning capacity. I expect that this inconsistency and the complexity of the rule explain the bad performance of CGC.

While CGC does not seem an advisable queueing system in the present study, further research is necessary to understand whether this also holds Further research could also focus on determining if there is a relation between different kinds of complexity of a queueing discipline and its functioning.

# Conclusion

The goal of this research was to answer the following question:

*How can the CG consistent be translated into a queueing discipline and how does it perform in comparison to Round-Robin and First Come First Serve?*

CGC falls under the Fair Queueing disciplines, and is another method of dividing capacity based on a rule from the Talmud with a different approach to fairness. CGC assigns server capacity based on the amount of jobs in that queue and contains two rules to ensure fairness. The idea of this research was to translate this into a new queueing discipline which would perform well were fairness is an important aspect.

The three queueing disciplines were built in a simulation, in which several scenarios were simulated in which the average arrivals of smallest job stream changed. The results show that the performance of CGC is not close to, and cannot compete with RR and FCFS. The performance was measured in two KPIs:

* Average time spent in queue per job (avgD)
* The coefficient of variation (CV) of jobs per job type

On both KPIs CGC performed worse than RR and FCFS. The lower the avgD and CV are the better the queueing discipline performed. Table 4. Average Sum of avgD and CV, shows the average sum of all the queues of all the scenarios.

|  |  |  |  |
| --- | --- | --- | --- |
|  | FCFS | RR | CGC |
| avgD | 9.935823 | 9.749073 | 10.24744 |
| CV | 2.107633 | 1.441481 | 2.635491 |

Table . Average Sum of avgD and CV

The bad performance of CGC might be attributed to its complex distribution of capacity in comparison to RR and FCFS. The CGC discipline is inconsistent because of the rules that lay at its root, expected is that this causes the bad performance in comparison to RR and FCFS.

At the start of the research the following hypotheses were posited:

*H1: CGC has a lower cycle time for the queue with the smallest job stream than the FCFS rule.*

This hypothesis was rejected since the results showed that the average cycle time of job stream A in FCFS was lower than that in CGC.

*H2: The CV of CGC has less variance amongst the queues in comparison to the CV of the RR queues.*

Hypothesis 2 was not rejected, the variance in CGC is spread more equal amongst the queues than in RR with in this research. However because of the high avgD and CV, CGC remains unable to even compete with RR. Thus CGC is unviable as a queueing discipline

The limitations in the design and study of CGC extended to, limited variability which is shown by not considering variability in capacity and neither possible failures nor scrapping of jobs. And in RR jobs could not be partially finished.

Further research towards CGC seems unadvisable, however the relationship between different kinds of complexity in a queueing discipline in relation to its performance might be a suitable subject.

# Bibliography

Alomari, F., & Menascé, D. A. (2014). Efficient Response Time Approximations for Multiclass Fork and Join Queues in Open and Closed Queuing Networks. *IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS*, 1437-1446.

Arpaci-Dusseau, R. H., & Arpaci-Dusseau, A. C. (2014). Scheduling: Introduction. In Arpaci-Dusseau, *Operating Systems* (pp. 1-12). Arpaci-Dusseau Books.

Aumann, R. J. (2002). Game Theory in the Talmud. *Research bulletin Series on Jewish Law and Economics*.

Azarfar, A., Frigon, J., & Sansò, B. (2012). Dynamic Selection of Priority Queueing Discipline in Cognitive Radio Networks. *Vehicular Technology Conference*, 1-5.

Bertrand, J., Wortmann, J., & Wijngaard, J. (1998). *Productiebeheersing en material management.* Educatieve Partners Nederland.

Demers, A., Keshav, S., & Shenker, S. (1989). Analysis and simulation of a fair queueing algorithm. *ACM SIGCOMM Computer Communication Review*, 3-26.

Goldberg, H. M. (1977). Analysis of the Earliest Due Date Scheduling Rule in Queueing Systems. *Mathematics of Operations Research,*, 145-154.

Hall, R. (1991). *Queueing Methods: For Services and Manufacturing.* Prentice Hall.

Hopp, W., & Spearman, M. (2008). *Factory Physics.* Singapore: McGraw-Hill.

Jackson, J. R. (1961). Queues with Dynamic Priority Discipline. *Management Science*, 18 - 34.

Kendall. (1953). Stochastic Processes Occurring in the Theory of Queues and their Analysis by the Method of the Imbedded Markov Chain. *The Annals of Mathematical Statistics*, 338-354.

Law, A. M. (2003). HOW TO CONDUCT A SUCCESSFULL SIMULATION STUDY. *Proceedings of the 2003 Winter Simulation Conference*.

Nagle, J. B. (1987). On Packet Switches with Infinite Storage. *IEEE Transactions on Communications*, 435-438.

Ouelhadj, D., & Petrovic, S. (2009). A survey of dynamic scheduling in manufacturing systems. *Journal of Scheduling*, 417 - 431.

Sargent, R. G. (2003). Verification and Validation of Simulation Models. *Proceedings of the 2003 Winter Simulation Conference*.

Schrage, L. (1968). Letter to the Editor - A Proof of the Optimality of the Shortest Remaining Processing Time Discipline. *Operations Research*, 687-690.

Sen, R. (2010). *Operations Research: Algorithms And Applications.* Prentice-Hall of India.

Sundarapandian, & V. (2009). Queueing Theory. In *Probability, Statistics and Queueing Theory.* PHI Learning.

# Appendix A. Model Testing

To follow is a list with tests which are performed with the code which is used. In some cases the results are shown, showing the results of all the test would be overwhelming. These test have been performed for all the different arrivals of the first queue.

* Build CGC according to the description of Aumann (2002), and test if the results identical to that represented by Aumann (2002).
* Check the workings of FCFS, RR, and CGC in a period.

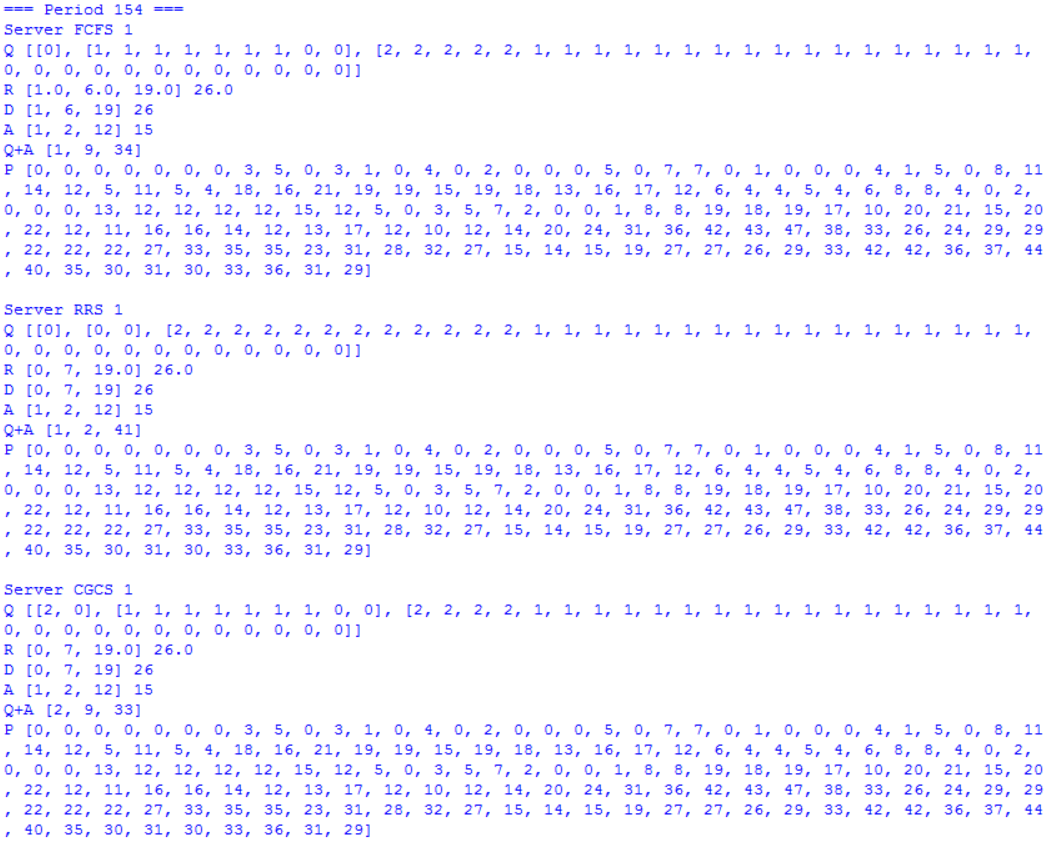


Figure . Results of random selected period

* Set capacity equal to arrivals to see queues fill.
* Change averages of arrival.
* Change number of queues.
* Compare queue length across queuing disciplines.
* Printing several code lines to check functioning.
* Comparison to the workings of simplified models.
* Change number of jobs in warm-up period.
* Measure the WIP and TH which should be equal across all queueing disciplines.

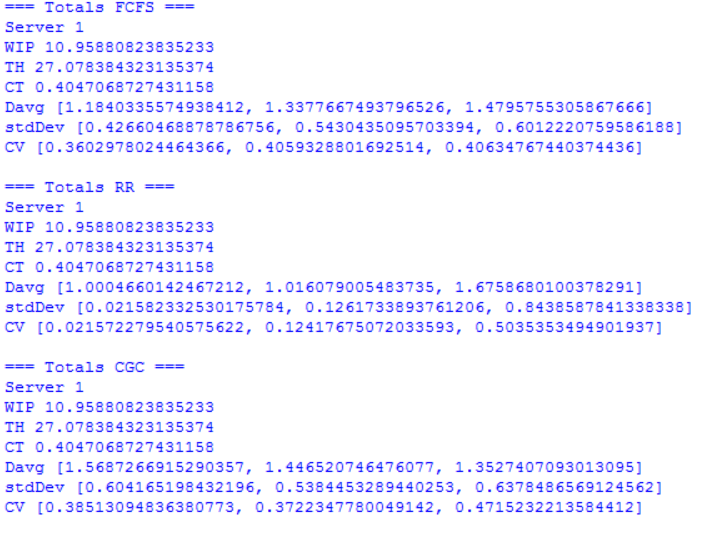
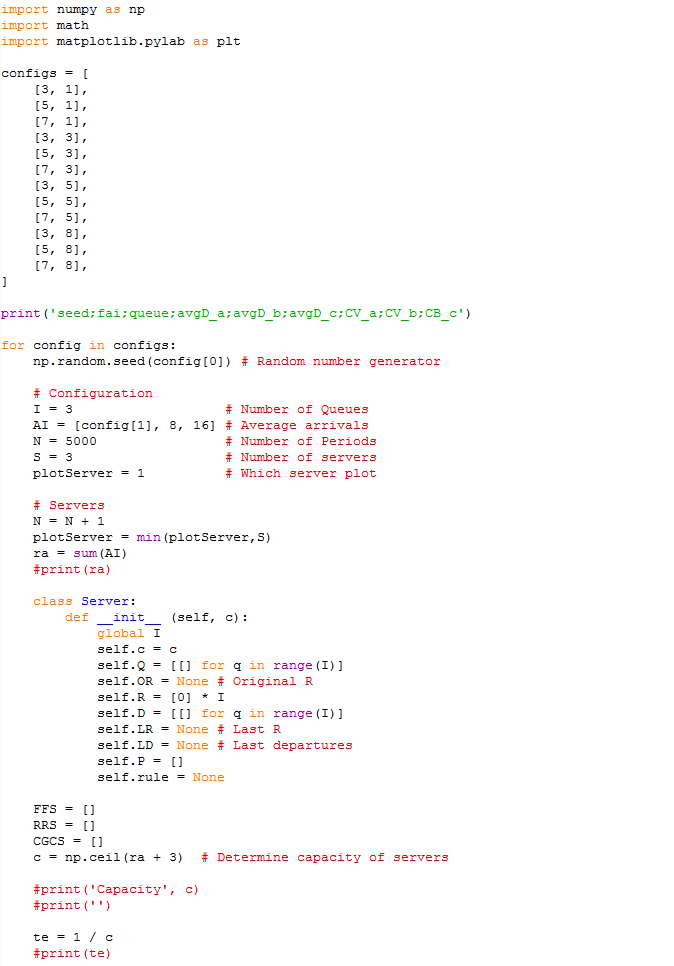
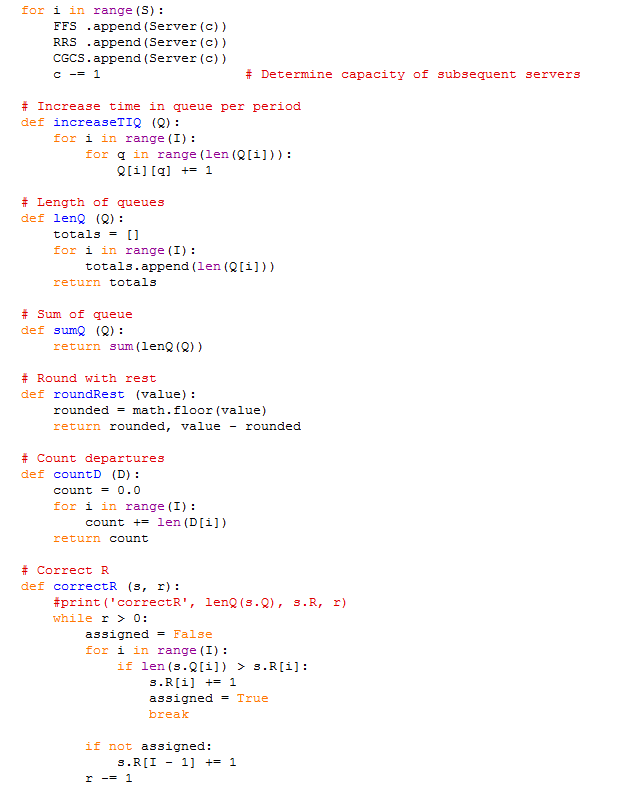
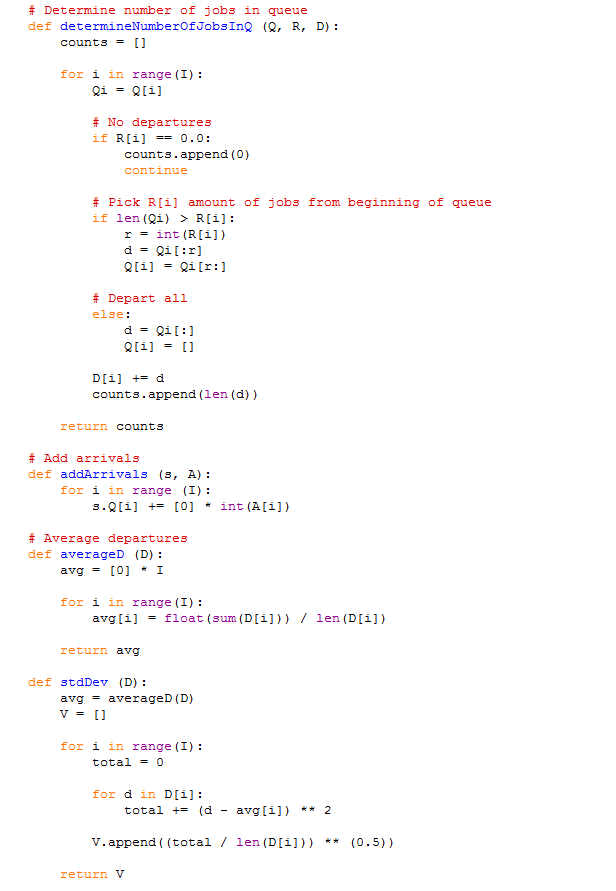


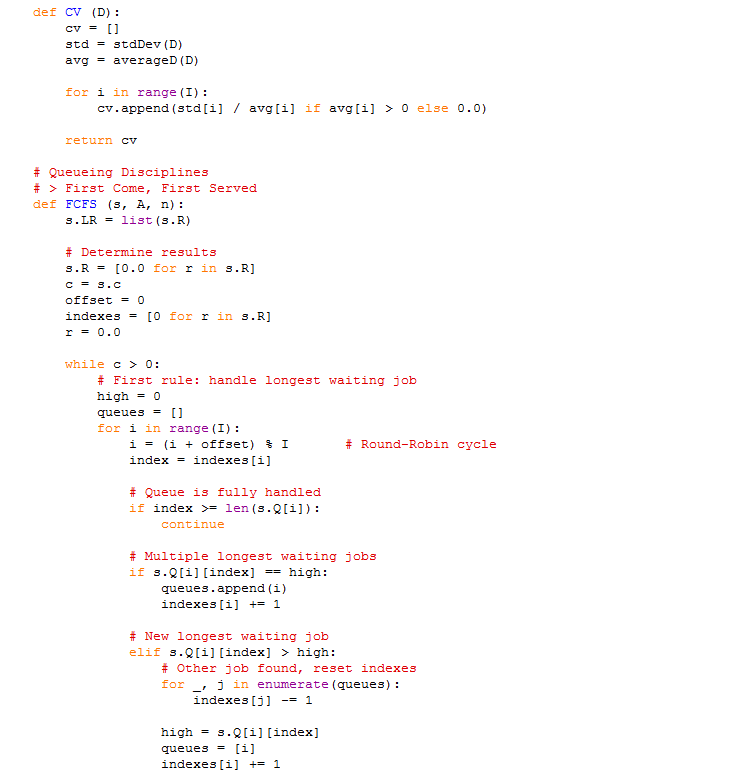
Figure . Results of a scenario

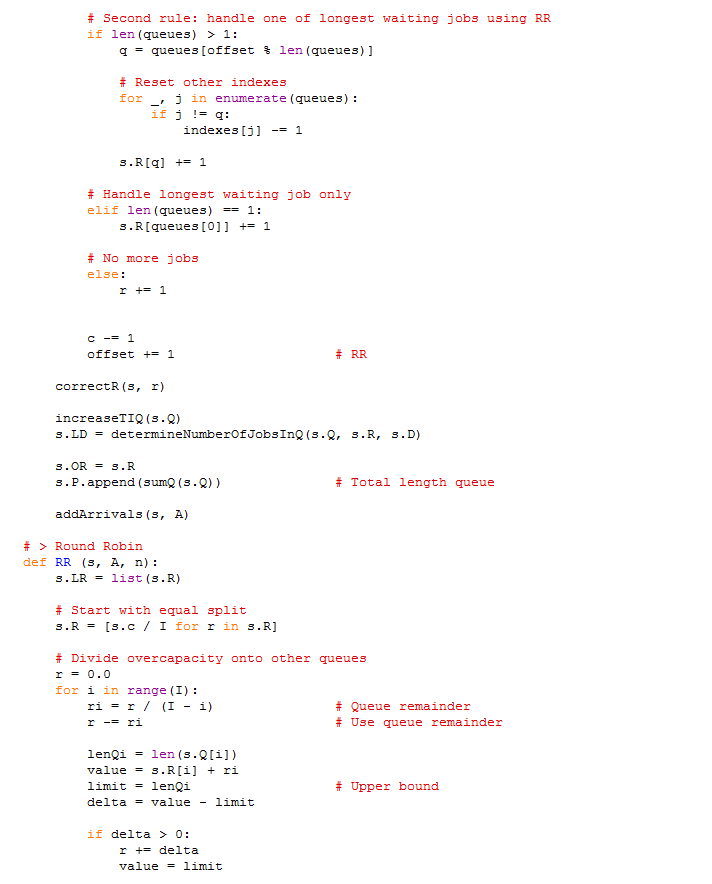
# Appendix B. Code

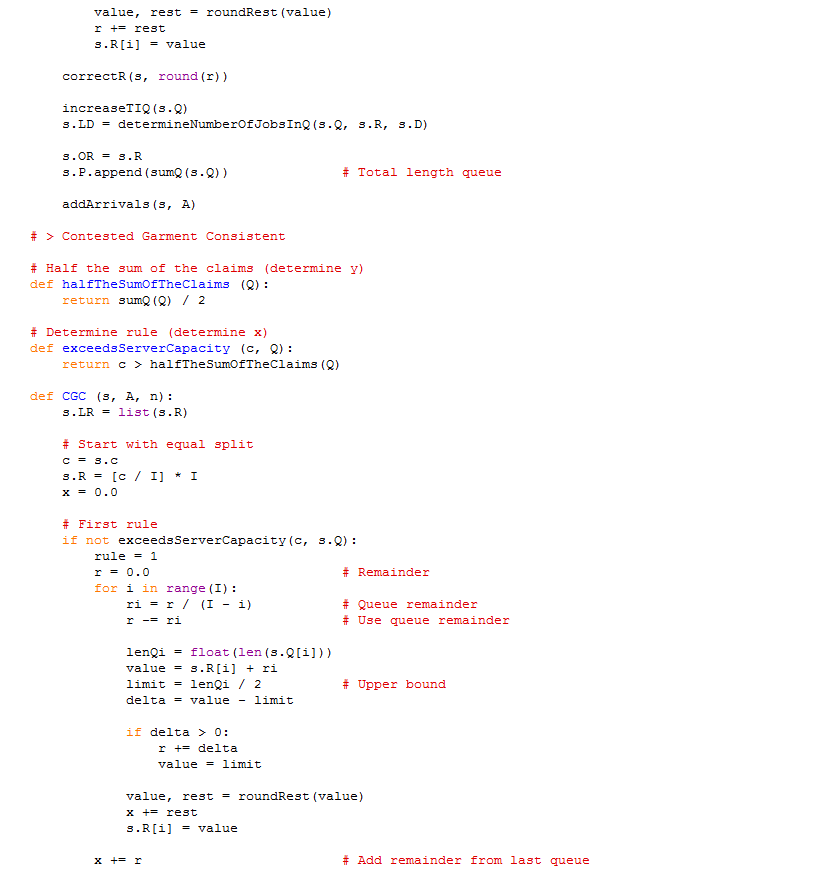


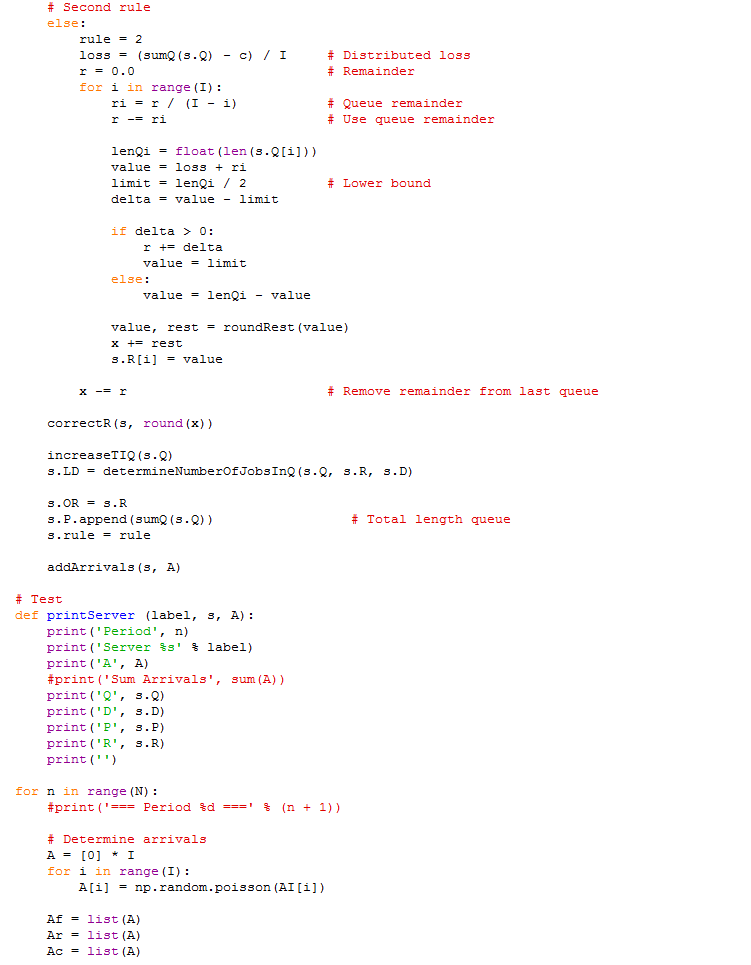












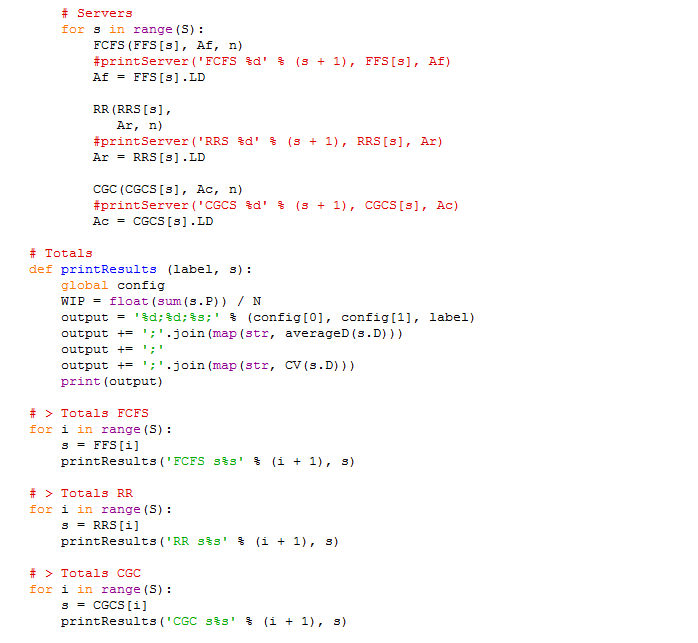


Figure . Code of the simulation

# Appendix C. Results

The results from the simulation are shown below. The first tables represent the averages of the seed values. The raw results represent the results which were extracted from the simulation.

## Average Scenario Values

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| A = 1 | avgD\_a | avgD\_b | avgD\_c | CV\_a | CV\_b | CB\_c |
| FCFS s1 | 1.001039 | 1.034758 | 1.10672 | 0.0289 | 0.176454 | 0.279126 |
| FCFS s2 | 1.000196 | 1.017783 | 1.098916 | 0.011297 | 0.129407 | 0.271544 |
| FCFS s3 | 1.039064 | 1.111068 | 1.281499 | 0.183349 | 0.28739 | 0.363579 |
| RR s1 | 1 | 1.006996 | 1.120701 | 0 | 0.082539 | 0.294751 |
| RR s2 | 1 | 1.004223 | 1.105725 | 0 | 0.064546 | 0.278368 |
| RR s3 | 1 | 1.005968 | 1.336593 | 0 | 0.07626 | 0.411451 |
| CGC s1 | 1.064131 | 1.10847 | 1.065849 | 0.231482 | 0.282836 | 0.232864 |
| CGC s2 | 1.045743 | 1.087386 | 1.061196 | 0.201194 | 0.260249 | 0.225762 |
| CGC s3 | 1.089909 | 1.2766 | 1.195418 | 0.267716 | 0.363572 | 0.351242 |

Table . Average results with arrivals [1,8,16]

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| A = 3 | avgD\_a | avgD\_b | avgD\_c | CV\_a | CV\_b | CB\_c |
| FCFS s1 | 1.004696 | 1.025833 | 1.123585 | 0.06757 | 0.154543 | 0.293222 |
| FCFS s2 | 1.003828 | 1.027293 | 1.110832 | 0.056517 | 0.156794 | 0.282252 |
| FCFS s3 | 1.087118 | 1.195094 | 1.325432 | 0.279092 | 0.36789 | 0.396794 |
| RR s1 | 1.000022 | 1.008068 | 1.133397 | 0.002699 | 0.088731 | 0.305375 |
| RR s2 | 1.000333 | 1.00444 | 1.123003 | 0.018181 | 0.066106 | 0.297144 |
| RR s3 | 1.000022 | 1.006523 | 1.436845 | 0.00272 | 0.079952 | 0.483727 |
| CGC s1 | 1.097028 | 1.112487 | 1.062696 | 0.272086 | 0.286787 | 0.22852 |
| CGC s2 | 1.084801 | 1.095815 | 1.06117 | 0.25872 | 0.270479 | 0.225595 |
| CGC s3 | 1.233625 | 1.308179 | 1.241245 | 0.348926 | 0.371894 | 0.426013 |

Table . Average results with arrivals [3,8,16]

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| A = 5 | avgD\_a | avgD\_b | avgD\_c | CV\_a | CV\_b | CB\_c |
| FCFS s1 | 1.014369 | 1.036045 | 1.133092 | 0.116053 | 0.180053 | 0.301286 |
| FCFS s2 | 1.009312 | 1.02064 | 1.126934 | 0.094123 | 0.138656 | 0.295271 |
| FCFS s3 | 1.146485 | 1.172575 | 1.35065 | 0.333055 | 0.35048 | 0.396467 |
| RR s1 | 1.003541 | 1.009427 | 1.149824 | 0.059113 | 0.095725 | 0.326198 |
| RR s2 | 1.000293 | 1.005114 | 1.137533 | 0.01705 | 0.070959 | 0.310753 |
| RR s3 | 1.000281 | 1.007443 | 1.47917 | 0.016282 | 0.085315 | 0.503894 |
| CGC s1 | 1.097892 | 1.117053 | 1.066379 | 0.272671 | 0.291174 | 0.235757 |
| CGC s2 | 1.08002 | 1.103329 | 1.063382 | 0.251746 | 0.277775 | 0.229275 |
| CGC s3 | 1.255582 | 1.312182 | 1.246589 | 0.35416 | 0.37337 | 0.429828 |

Table . Average results with arrivals [5,8,16]

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| A = 8 | avgD\_a | avgD\_b | avgD\_c | CV\_a | CV\_b | CB\_c |
| FCFS s1 | 1.042352 | 1.039233 | 1.134834 | 0.192836 | 0.186568 | 0.301064 |
| FCFS s2 | 1.032547 | 1.037665 | 1.129805 | 0.170818 | 0.1825 | 0.297228 |
| FCFS s3 | 1.189189 | 1.176392 | 1.356418 | 0.358514 | 0.353251 | 0.396589 |
| RR s1 | 1.02924 | 1.009259 | 1.1565 | 0.163616 | 0.094896 | 0.326108 |
| RR s2 | 1.004958 | 1.006779 | 1.159193 | 0.069652 | 0.081466 | 0.330162 |
| RR s3 | 1.004243 | 1.00527 | 1.535364 | 0.064567 | 0.072007 | 0.52561 |
| CGC s1 | 1.082637 | 1.127953 | 1.069997 | 0.254857 | 0.298667 | 0.238923 |
| CGC s2 | 1.070089 | 1.117238 | 1.070941 | 0.238317 | 0.290707 | 0.239743 |
| CGC s3 | 1.232846 | 1.323286 | 1.260633 | 0.345797 | 0.372525 | 0.440733 |

Table . Average results with arrivals [8,8,16]

## Raw Results

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| seed | fai | queue | avgD\_a | avgD\_b | avgD\_c | CV\_a | CV\_b | CB\_c |
| 3 | 1 | FCFS s1 | 1.0006 | 1.0366 | 1.1077 | 0.0248 | 0.1812 | 0.2803 |
| 3 | 1 | FCFS s2 | 1.0002 | 1.0175 | 1.0999 | 0.0143 | 0.1289 | 0.2726 |
| 3 | 1 | FCFS s3 | 1.0596 | 1.1412 | 1.3155 | 0.2235 | 0.3156 | 0.3740 |
| 3 | 1 | RR s1 | 1.0000 | 1.0067 | 1.1227 | 0.0000 | 0.0808 | 0.2976 |
| 3 | 1 | RR s2 | 1.0000 | 1.0040 | 1.1067 | 0.0000 | 0.0629 | 0.2797 |
| 3 | 1 | RR s3 | 1.0000 | 1.0053 | 1.3873 | 0.0000 | 0.0726 | 0.4329 |
| 3 | 1 | CGC s1 | 1.0645 | 1.1104 | 1.0668 | 0.2324 | 0.2851 | 0.2345 |
| 3 | 1 | CGC s2 | 1.0486 | 1.0882 | 1.0615 | 0.2059 | 0.2614 | 0.2264 |
| 3 | 1 | CGC s3 | 1.0916 | 1.3108 | 1.2285 | 0.2726 | 0.3708 | 0.3732 |
| 5 | 1 | FCFS s1 | 1.0023 | 1.0399 | 1.1122 | 0.0478 | 0.1883 | 0.2840 |
| 5 | 1 | FCFS s2 | 1.0004 | 1.0213 | 1.1045 | 0.0196 | 0.1415 | 0.2770 |
| 5 | 1 | FCFS s3 | 1.0319 | 1.1072 | 1.2773 | 0.1723 | 0.2892 | 0.3671 |
| 5 | 1 | RR s1 | 1.0000 | 1.0084 | 1.1282 | 0.0000 | 0.0905 | 0.3019 |
| 5 | 1 | RR s2 | 1.0000 | 1.0046 | 1.1130 | 0.0000 | 0.0673 | 0.2848 |
| 5 | 1 | RR s3 | 1.0000 | 1.0075 | 1.3297 | 0.0000 | 0.0855 | 0.4144 |
| 5 | 1 | CGC s1 | 1.0630 | 1.1144 | 1.0706 | 0.2300 | 0.2888 | 0.2398 |
| 5 | 1 | CGC s2 | 1.0464 | 1.0937 | 1.0650 | 0.2028 | 0.2671 | 0.2315 |
| 5 | 1 | CGC s3 | 1.0879 | 1.2695 | 1.1917 | 0.2646 | 0.3619 | 0.3564 |
| 7 | 1 | FCFS s1 | 1.0002 | 1.0277 | 1.1003 | 0.0141 | 0.1598 | 0.2731 |
| 7 | 1 | FCFS s2 | 1.0000 | 1.0145 | 1.0923 | 0.0000 | 0.1178 | 0.2650 |
| 7 | 1 | FCFS s3 | 1.0257 | 1.0848 | 1.2517 | 0.1542 | 0.2573 | 0.3496 |
| 7 | 1 | RR s1 | 1.0000 | 1.0059 | 1.1112 | 0.0000 | 0.0763 | 0.2847 |
| 7 | 1 | RR s2 | 1.0000 | 1.0041 | 1.0975 | 0.0000 | 0.0634 | 0.2706 |
| 7 | 1 | RR s3 | 1.0000 | 1.0051 | 1.2928 | 0.0000 | 0.0707 | 0.3871 |
| 7 | 1 | CGC s1 | 1.0649 | 1.1006 | 1.0602 | 0.2321 | 0.2747 | 0.2243 |
| 7 | 1 | CGC s2 | 1.0422 | 1.0802 | 1.0571 | 0.1948 | 0.2523 | 0.2194 |
| 7 | 1 | CGC s3 | 1.0902 | 1.2494 | 1.1660 | 0.2659 | 0.3580 | 0.3242 |
|  |  |  |  |  |  |  |  |  |
| 3 | 3 | FCFS s1 | 1.0059 | 1.0285 | 1.1233 | 0.0763 | 0.1621 | 0.2937 |
| 3 | 3 | FCFS s2 | 1.0030 | 1.0279 | 1.1103 | 0.0545 | 0.1603 | 0.2821 |
| 3 | 3 | FCFS s3 | 1.0779 | 1.1871 | 1.3117 | 0.2658 | 0.3511 | 0.3769 |
| 3 | 3 | RR s1 | 1.0000 | 1.0081 | 1.1347 | 0.0000 | 0.0889 | 0.3105 |
| 3 | 3 | RR s2 | 1.0004 | 1.0044 | 1.1226 | 0.0200 | 0.0662 | 0.2962 |
| 3 | 3 | RR s3 | 1.0001 | 1.0061 | 1.4173 | 0.0082 | 0.0773 | 0.4583 |
| 3 | 3 | CGC s1 | 1.0974 | 1.1124 | 1.0639 | 0.2734 | 0.2883 | 0.2312 |
| 3 | 3 | CGC s2 | 1.0851 | 1.0956 | 1.0609 | 0.2588 | 0.2707 | 0.2254 |
| 3 | 3 | CGC s3 | 1.2337 | 1.3112 | 1.2200 | 0.3491 | 0.3731 | 0.3893 |
| 5 | 3 | FCFS s1 | 1.0049 | 1.0263 | 1.1268 | 0.0692 | 0.1559 | 0.2953 |
| 5 | 3 | FCFS s2 | 1.0077 | 1.0361 | 1.1230 | 0.0866 | 0.1801 | 0.2927 |
| 5 | 3 | FCFS s3 | 1.1466 | 1.2638 | 1.3979 | 0.3898 | 0.4501 | 0.4604 |
| 5 | 3 | RR s1 | 1.0001 | 1.0084 | 1.1368 | 0.0081 | 0.0905 | 0.3063 |
| 5 | 3 | RR s2 | 1.0003 | 1.0050 | 1.1402 | 0.0162 | 0.0700 | 0.3167 |
| 5 | 3 | RR s3 | 1.0000 | 1.0069 | 1.5559 | 0.0000 | 0.0820 | 0.5961 |
| 5 | 3 | CGC s1 | 1.1003 | 1.1151 | 1.0637 | 0.2748 | 0.2882 | 0.2296 |
| 5 | 3 | CGC s2 | 1.0943 | 1.1092 | 1.0695 | 0.2703 | 0.2856 | 0.2384 |
| 5 | 3 | CGC s3 | 1.2518 | 1.3410 | 1.3387 | 0.3536 | 0.3786 | 0.5631 |
| 7 | 3 | FCFS s1 | 1.0033 | 1.0227 | 1.1206 | 0.0573 | 0.1456 | 0.2906 |
| 7 | 3 | FCFS s2 | 1.0008 | 1.0178 | 1.0992 | 0.0284 | 0.1300 | 0.2719 |
| 7 | 3 | FCFS s3 | 1.0368 | 1.1344 | 1.2668 | 0.1817 | 0.3025 | 0.3531 |
| 7 | 3 | RR s1 | 1.0000 | 1.0077 | 1.1287 | 0.0000 | 0.0869 | 0.2994 |
| 7 | 3 | RR s2 | 1.0003 | 1.0039 | 1.1062 | 0.0184 | 0.0621 | 0.2785 |
| 7 | 3 | RR s3 | 1.0000 | 1.0066 | 1.3373 | 0.0000 | 0.0806 | 0.3968 |
| 7 | 3 | CGC s1 | 1.0934 | 1.1100 | 1.0605 | 0.2680 | 0.2839 | 0.2248 |
| 7 | 3 | CGC s2 | 1.0751 | 1.0827 | 1.0531 | 0.2470 | 0.2551 | 0.2129 |
| 7 | 3 | CGC s3 | 1.2153 | 1.2723 | 1.1650 | 0.3441 | 0.3639 | 0.3256 |
|  |  |  |  |  |  |  |  |  |
| 3 | 5 | FCFS s1 | 1.0196 | 1.0413 | 1.1411 | 0.1369 | 0.1920 | 0.3084 |
| 3 | 5 | FCFS s2 | 1.0090 | 1.0202 | 1.1327 | 0.0934 | 0.1378 | 0.2995 |
| 3 | 5 | FCFS s3 | 1.1794 | 1.2070 | 1.3842 | 0.3482 | 0.3649 | 0.4012 |
| 3 | 5 | RR s1 | 1.0040 | 1.0092 | 1.1621 | 0.0626 | 0.0945 | 0.3425 |
| 3 | 5 | RR s2 | 1.0004 | 1.0052 | 1.1429 | 0.0189 | 0.0718 | 0.3137 |
| 3 | 5 | RR s3 | 1.0003 | 1.0073 | 1.5407 | 0.0178 | 0.0844 | 0.5141 |
| 3 | 5 | CGC s1 | 1.1039 | 1.1241 | 1.0731 | 0.2791 | 0.2982 | 0.2478 |
| 3 | 5 | CGC s2 | 1.0828 | 1.1078 | 1.0655 | 0.2557 | 0.2821 | 0.2322 |
| 3 | 5 | CGC s3 | 1.2766 | 1.3435 | 1.2851 | 0.3591 | 0.3792 | 0.4376 |
| 5 | 5 | FCFS s1 | 1.0145 | 1.0357 | 1.1358 | 0.1179 | 0.1798 | 0.3028 |
| 5 | 5 | FCFS s2 | 1.0127 | 1.0256 | 1.1360 | 0.1107 | 0.1543 | 0.3026 |
| 5 | 5 | FCFS s3 | 1.1543 | 1.1825 | 1.3669 | 0.3553 | 0.3708 | 0.4029 |
| 5 | 5 | RR s1 | 1.0036 | 1.0094 | 1.1523 | 0.0595 | 0.0954 | 0.3265 |
| 5 | 5 | RR s2 | 1.0003 | 1.0049 | 1.1502 | 0.0167 | 0.0698 | 0.3249 |
| 5 | 5 | RR s3 | 1.0001 | 1.0076 | 1.5026 | 0.0109 | 0.0863 | 0.5194 |
| 5 | 5 | CGC s1 | 1.1017 | 1.1199 | 1.0664 | 0.2767 | 0.2929 | 0.2354 |
| 5 | 5 | CGC s2 | 1.0882 | 1.1131 | 1.0687 | 0.2617 | 0.2882 | 0.2379 |
| 5 | 5 | CGC s3 | 1.2724 | 1.3298 | 1.2563 | 0.3572 | 0.3748 | 0.4594 |
| 7 | 5 | FCFS s1 | 1.0089 | 1.0311 | 1.1224 | 0.0933 | 0.1684 | 0.2926 |
| 7 | 5 | FCFS s2 | 1.0062 | 1.0161 | 1.1121 | 0.0782 | 0.1239 | 0.2837 |
| 7 | 5 | FCFS s3 | 1.1057 | 1.1282 | 1.3009 | 0.2956 | 0.3158 | 0.3853 |
| 7 | 5 | RR s1 | 1.0031 | 1.0098 | 1.1350 | 0.0552 | 0.0973 | 0.3096 |
| 7 | 5 | RR s2 | 1.0002 | 1.0052 | 1.1195 | 0.0156 | 0.0713 | 0.2937 |
| 7 | 5 | RR s3 | 1.0004 | 1.0074 | 1.3942 | 0.0201 | 0.0853 | 0.4782 |
| 7 | 5 | CGC s1 | 1.0881 | 1.1072 | 1.0597 | 0.2622 | 0.2825 | 0.2241 |
| 7 | 5 | CGC s2 | 1.0690 | 1.0891 | 1.0560 | 0.2378 | 0.2630 | 0.2177 |
| 7 | 5 | CGC s3 | 1.2177 | 1.2632 | 1.1984 | 0.3462 | 0.3660 | 0.3925 |
|  |  |  |  |  |  |  |  |  |
| 3 | 8 | FCFS s1 | 1.0487 | 1.0478 | 1.1489 | 0.2057 | 0.2049 | 0.3112 |
| 3 | 8 | FCFS s2 | 1.0382 | 1.0458 | 1.1445 | 0.1846 | 0.1998 | 0.3072 |
| 3 | 8 | FCFS s3 | 1.2288 | 1.2184 | 1.4023 | 0.3889 | 0.3880 | 0.4115 |
| 3 | 8 | RR s1 | 1.0306 | 1.0093 | 1.1775 | 0.1671 | 0.0952 | 0.3451 |
| 3 | 8 | RR s2 | 1.0050 | 1.0072 | 1.1807 | 0.0702 | 0.0837 | 0.3446 |
| 3 | 8 | RR s3 | 1.0044 | 1.0055 | 1.6224 | 0.0661 | 0.0738 | 0.5585 |
| 3 | 8 | CGC s1 | 1.0943 | 1.1419 | 1.0785 | 0.2682 | 0.3098 | 0.2516 |
| 3 | 8 | CGC s2 | 1.0823 | 1.1320 | 1.0788 | 0.2550 | 0.3030 | 0.2497 |
| 3 | 8 | CGC s3 | 1.2673 | 1.3646 | 1.3091 | 0.3551 | 0.3768 | 0.4893 |
| 5 | 8 | FCFS s1 | 1.0427 | 1.0368 | 1.1286 | 0.1938 | 0.1816 | 0.2966 |
| 5 | 8 | FCFS s2 | 1.0358 | 1.0385 | 1.1284 | 0.1793 | 0.1852 | 0.2972 |
| 5 | 8 | FCFS s3 | 1.2169 | 1.2003 | 1.3798 | 0.3916 | 0.3839 | 0.4160 |
| 5 | 8 | RR s1 | 1.0302 | 1.0092 | 1.1488 | 0.1660 | 0.0945 | 0.3195 |
| 5 | 8 | RR s2 | 1.0059 | 1.0062 | 1.1596 | 0.0762 | 0.0781 | 0.3355 |
| 5 | 8 | RR s3 | 1.0048 | 1.0050 | 1.5844 | 0.0691 | 0.0702 | 0.5687 |
| 5 | 8 | CGC s1 | 1.0799 | 1.1221 | 1.0670 | 0.2524 | 0.2941 | 0.2344 |
| 5 | 8 | CGC s2 | 1.0709 | 1.1163 | 1.0717 | 0.2404 | 0.2914 | 0.2417 |
| 5 | 8 | CGC s3 | 1.2446 | 1.3333 | 1.2991 | 0.3508 | 0.3769 | 0.4895 |
| 7 | 8 | FCFS s1 | 1.0356 | 1.0331 | 1.1269 | 0.1790 | 0.1733 | 0.2954 |
| 7 | 8 | FCFS s2 | 1.0237 | 1.0288 | 1.1164 | 0.1486 | 0.1625 | 0.2873 |
| 7 | 8 | FCFS s3 | 1.1219 | 1.1105 | 1.2871 | 0.2951 | 0.2879 | 0.3623 |
| 7 | 8 | RR s1 | 1.0270 | 1.0093 | 1.1432 | 0.1577 | 0.0950 | 0.3138 |
| 7 | 8 | RR s2 | 1.0040 | 1.0070 | 1.1373 | 0.0625 | 0.0826 | 0.3104 |
| 7 | 8 | RR s3 | 1.0035 | 1.0053 | 1.3993 | 0.0585 | 0.0721 | 0.4497 |
| 7 | 8 | CGC s1 | 1.0737 | 1.1199 | 1.0645 | 0.2440 | 0.2921 | 0.2307 |
| 7 | 8 | CGC s2 | 1.0571 | 1.1035 | 1.0624 | 0.2195 | 0.2776 | 0.2278 |
| 7 | 8 | CGC s3 | 1.1867 | 1.2720 | 1.1738 | 0.3315 | 0.3639 | 0.3434 |

Table . Raw results from the simulation

1. The Talmud is a central text in Judaism [↑](#footnote-ref-1)
2. In the Python code FCFS is modelled in three queues but treated as one queue. [↑](#footnote-ref-4)
3. The arrivals are shown, as for example, like [1,8,16]. This means the first queue has an average arrival rate of 1 job per period, the second queue 8, and the third queue 16. [↑](#footnote-ref-5)