Chapter 2 Exercises

2-1. For the hand simulation of the simple processing system, define another time-persistent statistic as the total number of parts in the system, including any parts in queue and in service. Augment Table 2-2 to track this as a new global variable, add new statistical accumulators to get its time average and maximum, and compute these values at the end.

A: Another part-persistent statistic *N(n)* was defined as the total number of parts in the system, involving those parts in queue and the part being worked on [*N(n)=Q(n)+B(n)*]. This equation is a state-variable as for every part that arrives, the part that is being worked on and the queue behind that part is checked. Similarly, the time-persistent statistic *N(t)* could be defined as a global variable that keeps track of when a part is or is not being worked on and what parts are then behind it [*N(t)=Q(t)+B(t)*]. For the hand simulation, the statistic accumulator of total parts in the system is now tracked. The time average and maximum of *N(n)* are also computed as seen in the table below. On average there are 2 parts in the system at any given time, and the maximum parts in the system was seen to be 4.

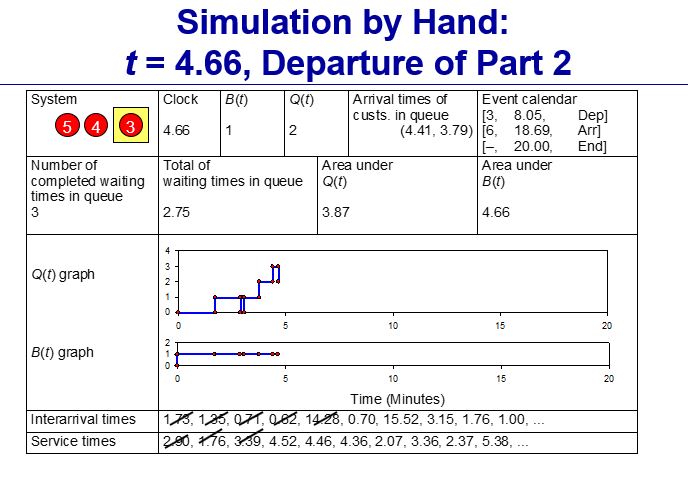
|  |  |  |  |
| --- | --- | --- | --- |
| Agent ID | B(n) | Q(n) | N(n) |
| 1 | 1 | 0 | 1 |
| 2 | 1 | 1 | 2 |
| 3 | 1 | 1 | 2 |
| 4 | 1 | 2 | 3 |
| 5 | 1 | 3 | 4 |
| 6 | 0 | 0 | 0 |
| 7 | 1 | 1 | 2 |
|  |  | Avg | 2 |
|  |  | Max | 4 |

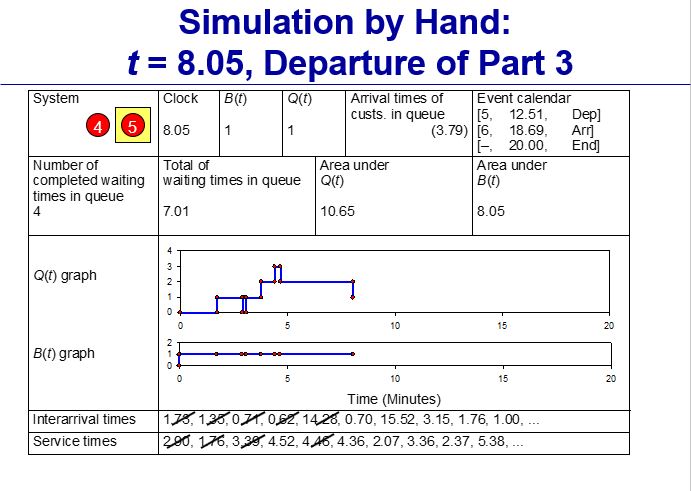
2-2. In the preceding exercise, did you really need to add state variables and keep track of new accumulators to get the *time-average* number of parts in the system? If not, why not? How about the *maximum* number of parts in the system?

A: In the previous exercise, having state variables was useful in seeing how many parts were in queue and were in the system when a new part arrived. However, for the calculation of the *time-average* and *maximum* number of parts in the system, the global variable accomplished this by itself. In determining the global and state variables, it was much easier to keep track of a part when it enters and leaves after being worked on versus checking at every time-state whether it was in the drill press.

2-3. In the hand simulation of the simple processing system, suppose that the *queue discipline* were changed so that when the drill press becomes idle and finds parts waiting in the queue, instead of taking the first one, it instead takes the one that will require the *shortest processing time* (this is sometimes called an *SPT* queue discipline). To make this work, you’ll need to assign a second attribute to parts in the system when they arrive, representing what their service time at the drill press will be. Redo the hand simulation. Is this a better rule? From what perspective?

A: For the hand simulation, the *shortest processing time* (*SPT*) queue discipline was implemented by considering the service time as the second attribute to parts in the system when they arrive (the first attribute being if there is a queue or not). In the hand simulation when Part 3 departed from the queue, Part 5 was serviced first as it had a shorted service time compared to Part 4.

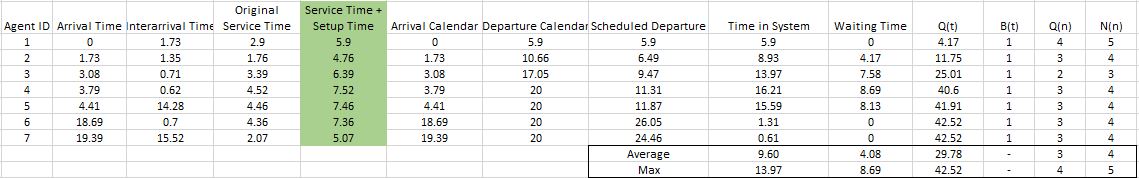




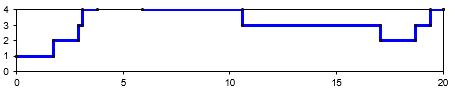
This rule helped in shortening the total queue time experienced in the system [*Q(t)*] as even though the number of parts in the queue [*Q(n)*] remained the same, the time in queue decreased from getting parts through the system faster. However when considering the individual queue time of Part 4 [*Q(4)*], the waiting time for that individual part was significantly longer as it was skipped in the queue. If parts continued to arrive with shorter service times than Part 4, it would keep getting skipped. If total queue time was not trying to be minimized but instead the average queue time of any one part in the system is, this would not be a great queuing model.

2-4. In the hand simulation of the simple processing system, suppose that a constant setup time of 3 minutes was required once a part entered the drill press but before its service could begin. When a setup is going on, regard the drill press as being busy. Plot the total number of parts in system as function of simulated time. As numerical output performance measures, report the total production (number of parts produced), the average and maximum waiting time in queue, the average and maximum total time in system, the time-average and maximum number of parts in queue, and the drill-press utilization. Redo the hand simulation and discuss the results.

A: For the hand simulation, a 3-minute setup time can easily be modeled as just increasing the service time of each part by an additional 3 minutes. This can be implemented because the drill press is regarded as being busy during this start up time. If the drill press was not considered being used during setup, as in the machine is not running, then another state-variable would have to be implemented. Without changes to the arrival time or interarrival time, the following hand simulation was done with the 3-minute setup time.



As most parts arrive in the first five minutes of the simulation, adding an additional 3-minute to the setup time of all parts, especially the first part, creates a queue backup that never dissipates. Total number of parts in the system as a function of time *N(t)* is given below. The large number of parts in the system at any given time is directly a cause from the queue having a backup.

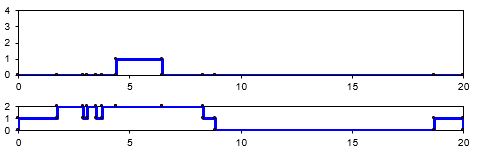


Compared to before when 5 parts was abled to be produced, now only a total of 3 parts was processed. The average and maximum waiting time in queue, the average and maximum total time in system, the time-average and maximum number of parts in queue, and the drill-press utilization are all given in the table above. The average and maximum total time in system are only given for the first three parts and not all parts as that would incorrectly lower the average, with the later parts not being processed at all by the end of the 20 minute simulation time.

2-5. In the hand simulation of the simple processing system, suppose the drill press can work on two parts simultaneously (and they enter, are processed, and leave the drill press independently). There’s no difference in processing speed if there are two parts in the drill press instead of one. Redefine *B(t)* to be the number of parts in service at the drill press at time (so 0≤*B(t)* ≤2), and the drill press utilization is redefined as

where *T=20*. Rerun the original simulation to measure the effect of this change.

A: The hand simulation was conducted again with two parts being abled to be processed simultaneously. This resulted in the following queue and drill utilization shown in the plots below.



|  |  |  |
| --- | --- | --- |
| Event Time | Q(n) | B(t) |
| 0.0000 | 0 | 1 |
| 1.7308 | 0 | 2 |
| 2.8956 | 0 | 1 |
| 3.0819 | 0 | 2 |
| 3.4900 | 0 | 1 |
| 3.7900 | 0 | 2 |
| 4.4098 | 1 | 2 |
| 6.4700 | 0 | 2 |
| 8.3100 | 0 | 1 |
| 8.8700 | 0 | 0 |
| 18.6900 | 0 | 1 |
| 20.0000 | 0 | 1 |

Total queue time as seen as the area underneath *Q(n)* drastically decreased. A queue was only realized for one part overall, with the addition of another part being able to be processed at any given time. Even though drill utilization greatly increased as seen in the are underneath *B(t)*, on average the utilization time per part can be found to not greatly change. Another way to interpret this study is if there were two drill units acting independently, taking parts in from the queue separate of the other drill press.