Report for discrete event simulation application

Yichao Yao

Chenwei Yue

·Brief Introduction

This program simulates the operation of a tandem queue and represents the process that manufacturing parts sequentially transfer to different stages(e.g station A, station B...) The program consists of the following several parts:

1)priority queue

Four functions were designed for the operation of this event priority queue: init_eList(); add_event(); pop_event(); delete_event();

When initializing the priority queue, the pFirst (an attribute of this event list) points to an event which works as a head node of this linked list.

The add_event() function was designed based on insertion sort algorithm. Every time a new event is added into this event queue, it will be inserted to a proper position based on its timestamp. Thus, the events in this priority queue are always sorted by ascending order.

2)FIFO(parts) queue

There are three functions to operate this FIFO queue: init_queue(); add_parts(); delete_parts();

When a part is added into this FIFO queue, it will be inserted in the last position of this queue.

3)simulation engine：

We use four integers to represent four types of events: arrival(int 1), departStation1(int 2), departStation2(int 3), departStation3(int 4).

During each iteration of the while-loop, the first event in the priority queue will be popped from the list. Then its event-handler function will be called, corresponding to this event’s type(e.g. arrival,departure...). Three FIFO queues, queueA queueB and queueC, for each station are established to store the part structure for each station. Subsequently, the current time will be updated and this event will be deleted(the storage for this event will also be released). And if the current time exceeds the end time, the while will end. And only the parts departing the system will be used in calculating time in system and queuing delay, all parts remaining in system will be deleted without considering the waiting_time.

4)Event handler

Four event handlers are defined in this library: Arrival(), departStation1(),departStation2() and departStation3();

Arrival(): Schedule next arrival event, generate new part and insert it to Queue A. In addition, this handler will check if station A is free, and schedule departStation1 event.

departStation1() & departStation2() both function schedule depart event for current station and next station, in addition, they move part in current FIFO queue to next FIFO queue.

departStation3() as final station in the system, this function will update wait_time and delete part in queueC. In addition it will schedule dapartStation3 event.

·Print out of a short simulation

Here we use a short simulation (end time equal 100) as an example to illustrate how this program works. Every time a new event occurs, the program will print the corresponding time, event type and the state of three stations.

```
Current Event: event type: 1, time: 94.610204
Current_time = 94.610204,and current queue size for each station A=0 B=1 C=3
Current eventlist is:
event type: 1, time: 94.610204
event type: 4, time: 96.440968
event type: 3, time: 103.692799

Current Event: event type: 4, time: 96.440968
Current_time = 96.440968,and current queue size for each station A=1 B=1 C=3
Current eventlist is:
event type: 4, time: 96.440968
event type: 1, time: 97.000596
event type: 3, time: 103.692799
event type: 2, time: 113.322734

Current Event: event type: 1, time: 97.000596
Current_time = 97.000596,and current queue size for each station A=1 B=1 C=2
Current eventlist is:
event type: 1, time: 97.000596
event type: 4, time: 102.373058
event type: 3, time: 103.692799
event type: 2, time: 113.322734

Current Event: event type: 4, time: 102.373058
Current_time = 102.373058,and current queue size for each station A=2 B=1 C=2
Current eventlist is:
event type: 4, time: 102.373058
event type: 3, time: 103.692799
event type: 1, time: 113.302497
event type: 2, time: 113.322734
```

Figure 1 Sample output for short simulation

From Figure 1, the program handles certain events well and the station queues are also under control. Besides, for a 100 unit time simulation, the program can stop at the correct point.

At the end of the simulation, the program will also print the average wait time and average time remaining in the system of one part.

```
Current Event: event type: 4, time: 102.373058
Current_time = 102.373058,and current queue size for each station A=2 B=1 C=2
Current eventlist is:
event type: 4, time: 102.373058
event type: 3, time: 103.692799
event type: 1, time: 113.302497
event type: 2, time: 113.322734

Arrival rate = 0.100000 ,Average time in system = 38.112870, Average queuing delay = 12.268252
```

Figure 2Finla result

From Figure 2, it's clear that the program can output the final result pretty well.

·Results for relation between arrival rate(1/A) and time

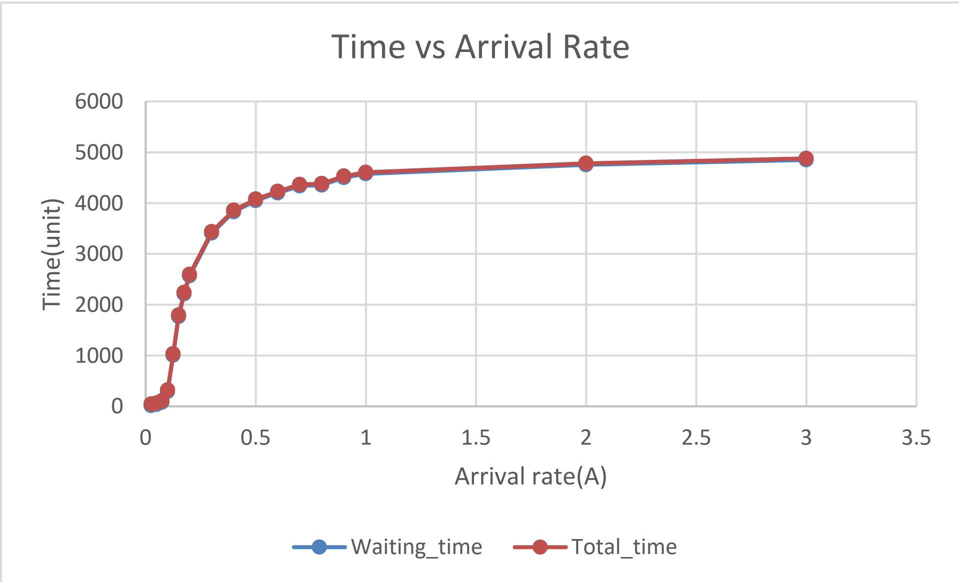

Figure 3 Time vs Arrival Rate

Figure 3 shows the relation between Arrival Rate and average time in system and average awaiting time. The difference between these two time is always 30 which is 3*service_time_mean. When the rate is low (< 0.1), the waiting_time is also extremely low, because in this condition, interarrival_time is much bigger than service time, all parts do not need to wait for long time. After 0,1, waiting time will grow with Arrival Rate until arrival rate reaches 1. In this period, service time and interarrival time is close, but the interarrival time is still bigger than service time, so that there will be stationary state of the queue size, and the waiting time will be determined by the arrival rate. After that waiting time increases slowly, since in this case, interarrival time is greater than service time, which means the size of queue will infinitely increase. But only the part passing the whole system will be considered in calculation of waiting time, so that even the queue is becoming longer, in period of 10000 unit time, the number of parts can be finished by this system is limited by a certain value. So that the arrival rate will not have huge effect on this system.