Determining the Most Efficient Number of Lines to Open in an Airport

Brandon Watkins
Department of Math and Science
Idaho State University
Pocatello, Idaho
watkbran@isu.edu

***Abstract*—This document describes the results of a simulator program that simulates the average time spent waiting in line by customers, with respect to the number of lines available and the time of day, or more specifically, the arrival rate of new customers.**

Keywords—airport, simulation, queue, linked queue, linked deque, linked stack)

I. Introduction

This document contains my findings for the CS2235 assignment "pa03." The purpose of this work was to find the smallest number of lines required to be kept open in an airport, while keeping customers satisfied with their time spent waiting in their respective lines, with respect to the time of day or arrival rate of customers. This was deemed necessary due to recent troubles balancing customer service and operating costs.

To determine the most efficient number of lines to have open, I was tasked with creating a simulator program that can accept the inputs: arrival rate, maximum number of available lines and number of iterations to find an average over, and output the average time customers spent in line, over a twelve hour period, disregarding those customers who are still in line at the end of the twelve hour period.

Additionally, I explored some of the differences between queues, deques and stacks, by using these to represent the lines. Upon beginning the simulation, I expected to see wait times reduce linearly relative to the number of open lines with more lines providing shorter wait times. I expected stacks to be slower, and I had no expectations for the deque.

II. Simulation Design

A. *Simulation Description*

The simulation measures the time customers spend waiting in line. The user runs the program with inputs representing the arrival rate, maximum number of lines available, and the number of iterations to find the average over, to fine-tune the output, which is the average number of minutes customers spent in your lines. This allows the user to determine the most efficient, while remaining effective, number of lines to open for different times of the day.

B. *How the Simulation Works*

Each of the line types, queue, deque and stack, are implemented using a doubly-linked list. For the queue, the arriving customers enter the back of the line, and leave through the front. For the deque, the arriving customers randomly choose whether to enter the front or the back of the line, and whichever customer has been waiting longer, between the front and the back of the line, or the front of the line if they are equal, will leave the line. For stack, the arriving customers both join and leave through the front of the line.

For each line type (queue, deque and stack), the simulation simulates the arrival of a semi-random number of new customers, who each take turns entering the currently shortest line, while two customers leave each of the lines, every minute, for a twelve hour period, for each of the potential number of lines, up to the maximum number of lines as dictated by the user, and repeated the user-specified number of times.

The time is recorded as each customer enters a line, and as they leave the line the current time minus the stored time equates to their total time spent in line, which is recorded to later be used to find the average time customers spent in line. Each of the user-defined iterations will result in an addition to the overall time spent waiting in line, to be averaged after each of the different scenarios representing different numbers of available lines, which is then output to the user.

C. *End Goals*

The goal was to increase my understanding of the differences, uses and implementations for some of the different data structures, specifically the queue, deque and stack, and to experiment with how the number of lines, arrival rate, and average wait times correlate. I suspect it was also to be a reminder that just because a certain data structure can theoretically work, doesn't mean that it should or will, for the particular situation.

III. Experimental Methods

A. The Simulation I/O

The simulator allows the user to decide how many new customers arrive every minute, how many lines they have available, how many times they wish to run the scenario to find the overall average, and a seed to be used for generating the pseudo-random number of arriving customers. Then it will output the average wait time for each set of lines, up the maximum stated by the user.

B. The Comparisons

I will be directly comparing the number of available lines to the average time spent waiting in line by the customers. This will exclude any customers still waiting in line at the end of the simulation.

C. The Experiment

The plan was to run the program with an arrival rate of eighteen customers per minute and a maximum of ten available lines, using fifty iterations for a solid average and a seed of one thousand twenty-four. Again, this will be over a twelve hour, or seven hundred twenty minute, time period. Then I can read the output to determine which number of available lines is most efficient, based on how long they were in the queue for.

Since I don't actually have a time requirement, or an acceptable range, I can't determine how many lines should be used, but I will instead find the relationship between time spent in line and the number of lines available.

IV. Results

A. Average Wait Time

As Table I shows, on average, there's a three hundred twenty minute wait time, assuming only one available line. Each additional line added reduces the average wait time by about forty minutes, indicating a linear relationship between average wait time and available lines, until the tenth available line, where the difference in average wait time is minor.

TABLE I. Available Lines and Effectiveness

Available Lines	**Average Wait Time (Minutes)**		
	Queue	**Deque**	**Stack**
1	319	0	0
2	279	0	0
3	239	0	0
4	199	0	0
5	159	8	0
6	119	31	0
7	79	38	0
8	39	27	0
9	2	2	1
10	0	0	0

B. Queues, Deques and Stacks

While the deque and stack wait times look ridiculously low, it is only because the people with the longest wait times are getting buried by the new people, and only the new people ever leave the line, until nine lines are available, skewing the results.

V. Analysis and Interpretation

Due to the linear relationship between available lines and average wait time, each additional available line has an identical value, until the tenth line. It seems to me that the time spent in lines relative to the number of lines open doesn't decrease until nine lines are opened. Opening the tenth line would essentially empty all lines, allowing just as many people to leave the line as are entering.

Having a single available line has a fifteen percent higher wait time than if they had opened a second. However, a single line has a sixteen hundred percent higher wait time than if they had nine open lines. The wait time decreases exponentially with each additional line.

VI. Conclusions

My original theses seems to be correct, the relationship between available lines and the average time spent waiting in line is linear. However, the stack-line technically has the lowest waiting times, due to not giving the earlier customers a chance to ever leave the line, and record their times. I suppose this is a good example of a data structure that theoretically can work, and may even look like it does a good job, but it doesn't really fulfill its purpose. The deque-line is essentially a hybrid between the other two, fulfilling the same lesson.

Due to the linear relationship between available lines and average wait time, if it's worth opening a second line, it's worth opening nine, since each additional line has an identical value (wait times reduced by forty minutes), until the tenth line. Whether it's worth opening a tenth line depends on the value the airport assigns to the minimal inconvenience of a two minute wait. The exponentially decreasing wait time mentioned above is yet another reason the airport should open nine or ten lines, versus just a couple.

In conclusion, anything under nine available lines would be foolish, opening a tenth would be worth considering if the additional operating costs are worth having shorter lines by two minutes.

References

[1] *IEEE eXpress Conference Publishing*, IEEE, 2020. Accessed on June 25, 2020. [Online]. Available: https://www.ieee.org/conferences/publishing/templates.html