Report

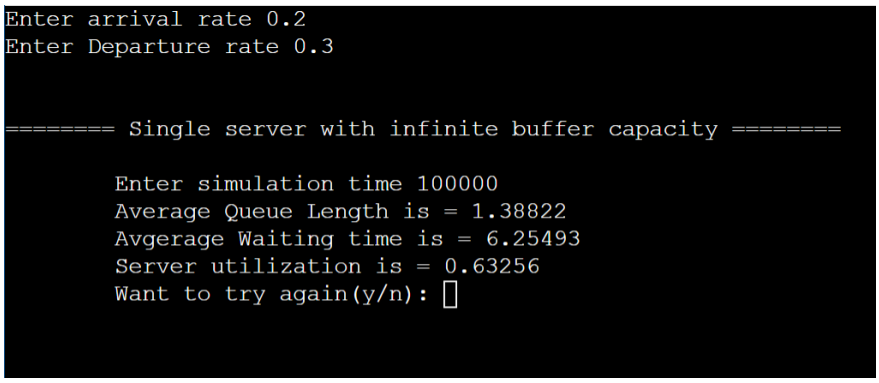
Ensuring the seamless flow of passengers through airport security lines is of paramount importance for both customer satisfaction and the efficient functioning of airport operations. This detailed report provides a thorough examination of diverse models and optimization techniques applicable to the security screening process at airports. The primary objective is to assess how varying arrival and departure rates influence key metrics such as average queue length, system utilization, and the overall quality of the passenger experience.

The airport security screening process was simulated using a C++ model, representing it as a queuing system with consideration for both single and multiple security lines. This simulation provided the flexibility to modify arrival and departure rates, enabling a comprehensive evaluation of their impact on the overall effectiveness of the security screening process. Essential performance indicators, including average waiting time, average queue length, and system utilization, were systematically gathered and thoroughly examined across different scenarios and optimization strategies.

Changing arrival and departure rates:

Run the simulations by changing the arrival rates by keeping departure rate same and again run the simulations by changing the departure rates by keeping arrival rate same

Start with arrival rate=0.2 and departure rate=0.3 and increase arrival rate by 0.1



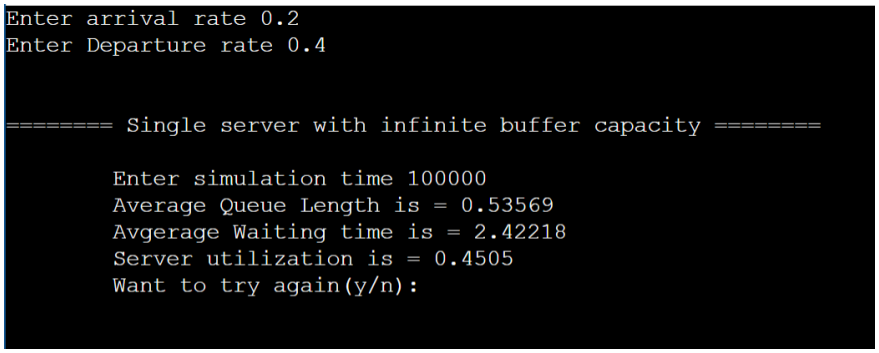
```
Enter arrival rate 0.3
Enter Departure rate 0.3


======== Single server with infinite buffer capacity ========

        Enter simulation time 100000
        Average Queue Length is = 296.563
        Avgerage Waiting time is = 846.671
        Server utilization is = 0.99951
        Want to try again(y/n):
```

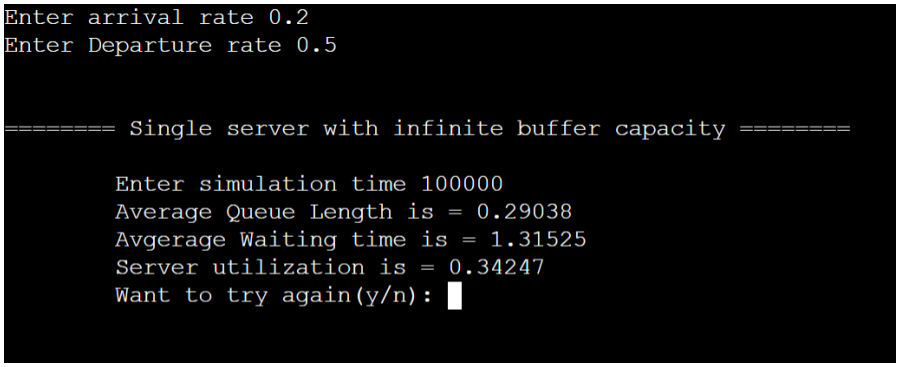
```
Enter arrival rate 0.4
Enter Departure rate 0.3


======== Single server with infinite buffer capacity ========

        Enter simulation time 100000
        Average Queue Length is = 7257.13
        Avgerage Waiting time is = 10307.1
        Server utilization is = 0.99992
        Want to try again(y/n):
```

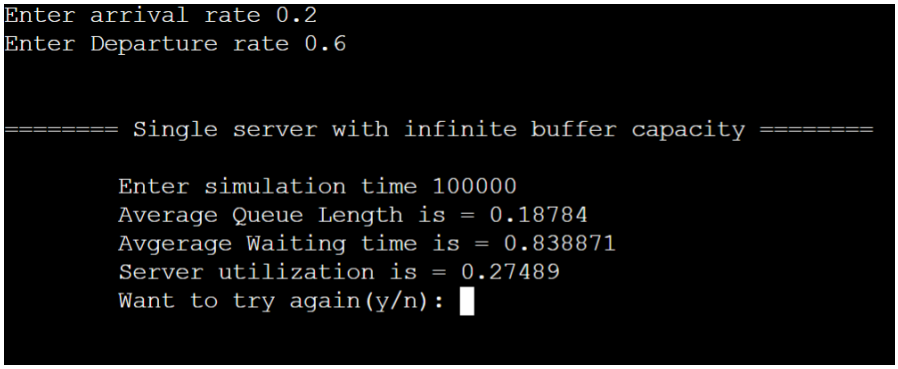
```
Enter arrival rate 0.5
Enter Departure rate 0.3


======== Single server with infinite buffer capacity ========

        Enter simulation time 100000
        Average Queue Length is = 14507.3
        Avgerage Waiting time is = 12497.5
        Server utilization is = 0.99997
        Want to try again(y/n):
```

Start with arrival rate=0.2 and departure rate=0.3 and increase departure rate by 0.1

```
Enter arrival rate 0.2
Enter Departure rate 0.3


======== Single server with infinite buffer capacity ========

        Enter simulation time 100000
        Average Queue Length is = 1.38822
        Avgerage Waiting time is = 6.25493
        Server utilization is = 0.63256
        Want to try again(y/n):
```







From above we can see that as the departure rates increases for a fixed lamda server utilization decreases whereas as lamda value is increasing server utilization is also increasing

→Conclusion:

So in order to have better server utilization take lamda value large (say 0.8) and then keep departure rate close to it(say 0.9) ensuring lamda is less than mu (stability constraint)

Changing buffer capacity :

We fix lamda and mu values as 0.8 and 0.9 as we have seen from above to have better server utilization

Now we gradually increased the buffer size value from 10 to 10,000 and found out that the changes were stark and significant between 10 and 160 ,pointing

to exponential or there-about increase,however after that the server utilisation values obtained evened out ;thus increasing our faith in the dependence being exponential. Thus we conclude that for the values of λ and μ ,that we took ;200(for safety margin) is a good buffer size,any increase beyond that in buffer size pays little dividends in server utilisation value.

```
Enter buffer size 10
Enter the simulating time 100000
Average Queue Length is = 0.00393
Avgerage Waiting time is = 2.31176
Server utilization is = 0.00111
Want to try again(y/n): y

Enter buffer size 50
Enter the simulating time 100000
Average Queue Length is = 0.20592
Avgerage Waiting time is = 6.86171
Server utilization is = 0.02132
Want to try again(y/n): y

Enter buffer size 100
Enter the simulating time 100000
Average Queue Length is = 10.0312
Avgerage Waiting time is = 8.23596
Server utilization is = 0.8358
Want to try again(y/n): y
```

```
Enter buffer size 200
Enter the simulating time 100000
Average Queue Length is = 11.6077
Avgerage Waiting time is = 9.38636
Server utilization is = 0.84607
Want to try again(y/n): y

Enter buffer size 300
Enter the simulating time 100000
Average Queue Length is = 10.8997
Avgerage Waiting time is = 8.88031
Server utilization is = 0.8422
Want to try again(y/n): y

Enter buffer size 1000
Enter the simulating time 100000
Average Queue Length is = 11.7934
Avgerage Waiting time is = 9.56277
Server utilization is = 0.84528
Want to try again(y/n): y
```

→Conclusion:

Instead of wasting buffer capacity or by considering it infinite we can keep it finite as beyond some value the server utilization is almost flattened.

Increasing number of servers:

As the servers increased server utilization and average queue length and waiting time decreases

Also the fall from n=1 to n=5 is sharp and there after the curve sort of evens out pointing to an inverse exponential relation between the server utilisation. For n=5 we find that the server utilisation is the smallest and thereafter evens out.

```
Enter simulating time:100000
Enter the number of servers 1
Average Queue Length is = 10.4108
Avgerage Waiting time is = 8.49219
Server utilization is = 0.83523
Want to try again(y/n): y

Enter simulating time:100000
Enter the number of servers 3
Average Queue Length is = 0.1589
Avgerage Waiting time is = 0.129197
Server utilization is = 0.3104
Want to try again(y/n): y

Enter simulating time:100000
Enter the number of servers 5
Average Queue Length is = 0.01207
Avgerage Waiting time is = 0.00977993
Server utilization is = 0.264155
Want to try again(y/n): y
```

```
Enter simulating time:100000
Enter the number of servers 7
Average Queue Length is = 0.00101
Avgerage Waiting time is = 0.000823817
Server utilization is = 0.259265
Want to try again(y/n): y

Enter simulating time:100000
Enter the number of servers 10
Average Queue Length is = 1e-05
Avgerage Waiting time is = 8.1614e-06
Server utilization is = 0.258295
Want to try again(y/n): y
```

→Conclusion:

Eventhough, increasing the number of servers benefits us in the form of decreasing average waiting time and average queue length there is no point in increasing it further as the decrease in the latter evens out

Changing buffer capacity in multi-server system:

In this also similar to above in the case of single server on increasing the buffer capacity initially the increase in server utility is sharp and thereafter it flattens out so we can say that having a buffer of size 25 is optimal. (lamda=0.8,mu=0.9, and servers=5 from above conclusions)

```
Enter simulating time:100000
Enter the number of servers 5
Enter the buffer size 10
Average Queue Length is = 8e-05
Avgerage Waiting time is = 0.00216041
Server utilization is = 0.00783
Want to try again(y/n): y

Enter simulating time:100000
Enter the number of servers 5
Enter the buffer size 20
Average Queue Length is = 0.00023
Avgerage Waiting time is = 0.00669383
Server utilization is = 0.0075
Want to try again(y/n): y
```

```
Enter simulating time:100000
Enter the number of servers 5
Enter the buffer size 30
Average Queue Length is = 0.01203
Avgerage Waiting time is = 0.00988578
Server utilization is = 0.261185
Want to try again(y/n): y

Enter simulating time:100000
Enter the number of servers 5
Enter the buffer size 50
Average Queue Length is = 0.01381
Avgerage Waiting time is = 0.0112446
Server utilization is = 0.2625
Want to try again(y/n):
```

Analysis:

Taking larger values of lamda and mu probably difference of 0.1-0.2 increases system utilization in M/M/1 model

Then having our airport 5 servers instead of 1 decreases avg waiting time and avg queue length

Then instead of having infinite buffer capacity we restricted it to 25 without much change in the total performance

Comprehensive Analysis:

-Impact of Arrival Rates on Queue Length and System Utilization:

The simulation results showed that when more passengers arrived at the security checkpoint, the lines got longer, and people had to wait longer. This also made the security process less efficient because the machines had more idle time. On the other hand, when fewer passengers arrived, the lines were shorter, and the security process ran more smoothly and quickly.

-Impact of Departure Rates on Queue Length and System Utilization:

The speed at which passengers moved through the security line, known as the departure rate, turned out to be a pivotal factor in how long they had to wait. When the departure rate was low, passengers had to wait longer, which led to frustration and overcrowding. Conversely, when the departure rate was higher, the flow through the security line was smoother, resulting in shorter waiting times and happier passengers overall.

-Impact of Buffer capacity:

Increasing the buffer size proved to be an effective strategy in minimizing the number of rejected passengers due to a full buffer. By accommodating a greater number of waiting passengers, the optimization strategy contributed to smoother operations and reduced passenger dissatisfaction, ultimately enhancing the overall airport experience but increasing buffer capacity beyond a point there is no much gain and infact the cost to have more buffer space will bw more so increasing beyond a limit is not cost efficient. Optimal buffer capacity has to be chosen based on the statistics of arrival and departure rate of passengers.

-Further optimisations:

Implementing dynamic staff allocation strategies based on real-time passenger flow data significantly improved resource utilization and queue management. By allocating staff according to fluctuating arrival rates and queue lengths,

airports could optimize security operations and ensure efficient utilization of personnel, leading to a more seamless and satisfactory passenger experience.

Based on the extensive analysis, it is recommended that airports consider the integration of advanced technologies such as artificial intelligence and machine learning in their security line management systems. Realtime data analysis and predictive modeling can enable proactive decision-making, leading to more efficient resource utilization and enhanced passenger satisfaction. Additionally, the implementation of automated queue management systems and self-service kiosks can further optimize the screening process, reducing wait times and improving overall operational efficiency.

Conclusion:

Above study highlights just how crucial it is to use smart strategies for making the airport security process run smoothly. By tweaking things like how many people arrive, how fast they move through, and how many staff members are on hand, airports can make sure everything runs well. This means shorter wait times and happier passengers. It's also important for airports to use modern technology and data analysis to adjust things in real-time and keep the security lines moving smoothly. In a nutshell, the study shows that taking proactive steps and using smart techniques can greatly improve the efficiency of airport security, making travel a lot more pleasant for everyone.

****