Original Research Article

Discrete-Event Simulation of Clients Flow in Ante-Natal Clinic

Abstract

Clients (expecting mothers) wait for hours in ante-natal clinic to receive medical service – waiting before, during or after being served. This study deals with a dynamic queuing system. Results of the study evaluate the effectiveness of queuing simulation model by identifying the ante-natal clinic queuing system parameters in terms of server utilization, usage, and clients flow time. The study uses the Simmer package in R for Discrete-event Simulation of the clients' flow in the system. The study showed that the resources are highly utilized with a bottleneck at the Doctors station, with constant service time for all clients, and long waiting time in the system. By replicating the parameters or replicate the model execution, once, with different initial conditions (by adding resources) and then performing another simulation over the output, the result showed that the resources are utilized with no bottlenecks at each server station, constant activity and flow time for all clients (expecting mothers). Hence, the model has proved to be accurate and efficient. This will help the clinic to utilize the resources and reduce long flow time.

Keyword: Discrete-event, Simulation, Clients' flow, Simmer Package, R

---

1. INTRODUCTION

The complexity of many real-world systems involves many unaffordable analytical models, and consequently, such systems are commonly studied using simulation. As defined by Shannon (1975), simulation "is the process of designing a model of a real system and conducting experiments with this model for either understanding the behaviour of the system or of evaluating various strategies for the operation of the system". Further, Hejase and Hejase (2012) contend that simulation "is a methodology that allows organizations to approach a business opportunity affected by risk virtually by creating an analogous situation without the commitment of scarce resources." (p. 446). Different types of simulation apply depending on the nature of the system under consideration. A common model taxonomy classifies a simulation problem along three main dimensions (Law and Kelton, 2000): (i) Deterministic vs. Stochastic. (ii). Static vs. Dynamic (depending on whether they require a time component). (iii). Continuous vs. Discrete (depending on how the system changes). Discrete-event simulation is a specific technique for modelling stochastic and or deterministic, dynamic and evolving system.

Simulation is useful to measure performance in systems that are so complex that they cannot be described by analytical queuing models (Green, 2006). To model any system we need to define its state space, that is, the variables that govern the behaviour of the system concerning the metrics being estimated. If the variables continuously change with time it is called a continuous system. If the system state instantaneously changes at discrete points in time, instead of continuously, it is called a discrete system.

1.1 Discrete Event Simulation

Discrete-event simulation is an analysis methodology that permits hospital administrators/clinic managers to evaluate the efforts of existing healthcare delivery systems, to ask "what if?" questions, and design new healthcare delivery system operations. Discrete-event simulation permits analysing bottlenecks and modelling the details of complex patients flow in hospitals (Jacobson et al., 2006), it

has become a popular tool for health care decision-makers to support their efforts in achieving their objectives.

Discrete- event simulation is also used as a forecasting tool to assess the potential impact of changes in patients' flow, to examine asset allocation needs (such as in staffing levels or physical capacity), and or to investigate the complex relationships among different system variables (such as the rate of patients' arrivals or the rate of patient service delivery). Such information allows healthcare administrators and analysts to identify management alternatives that can be used to reconfigure existing healthcare systems, to improve system performance or design, and or to plan new systems, without altering the existing system.

Discrete-event simulation in the analysis of healthcare systems has become increasingly more accepted by healthcare decision-makers as a viable tool for improving operations (Jacobson et al., 2006).

This study aims to use Discrete-event simulation model to minimize the waiting time of expecting mothers and maximize the utilization of the resources (Nurse, Doctor, Administrator) in ante-natal care unit of Aminu Kano Teaching Hospital.

2. MATERIALS AND METHOD

2.1 Method of Data Collection

Primary data collection was performed using observation method which involves the physical presence of researchers collecting the data. According to Hejase and Hejase (2013), "observation methods rely on watching and examining people's behaviours within certain predefined settings. Observational data is usually collected without the researcher's direct intervention" (p. 115). The researcher observed the arrivals and departures of expecting mothers at the Ante-Natal Care Unit per hour, including the service time at each phase.

2.2 Methodologies of Discrete-event Simulation

We now consider how to simulate discrete-event systems. The most common strategies for designing and implementing discrete-event simulation programs are (Mansharamani, 1997)

1. Event Scheduling: In the event schedule, list of all events in the system are constructed. Each event is taken individually and described in terms of the particular interaction between entity and resource. Associated with each event is the corresponding action or procedure to be invoked when the event occurs.
2. Activity Scanning: The purpose of the activity scanning is to overcome the reactivation problem of event scheduling.
3. Process Interaction: The process interaction methodology describes the system's workings from the view-point of an entity flowing through the system.
4. System State Variables: The system state variables are a set of data required to define the internal process within the system at a given point of time.

2.3 Discrete-event Simulation Modelling Concepts

1. Model: A model is a representation of an actual system.

2. Entities and attributes: An entity represents an object whose value can be static or dynamic, depending upon the process with other entities. Attributes are the local values used by the entity.

3. Resources: A resource is an entity that provides service to one or more dynamic entities at a time.

4. List Processing: Entities are managed by allocating them to resources that provide service, by attaching them to event notices thereby suspending their activity into the future, or by placing them into an ordered list. Lists are used to represent the queues by the entities and resources.

5. Activities and Delays: An activity is a duration time whose duration is known before the commencement of the activity whereas a delay is an indefinite duration that is caused by some combination of system conditions.

6. Events: An event is any change in the state of the system.

2.4 Steps in a Simulation Study

Modelling the process include the following steps

Step I: Examine the problem. In this stage, the simulation analyst must understand the problem and choose its classification accordingly, such as deterministic or stochastic.

Step II: Design a model. In this stage, the simulator will perform the following tasks which will help to design a model –

- Collect data as per the system behaviour and future requirements.
- Analyse the system features, its assumptions and necessary actions to be taken to make the model successful.
- Determine the variable names, functions, units, relationships and their applications used in the model.
- Solve the model using a suitable technique and verify the result using verification methods. Next, validate the result.
- Prepare a report which includes results, interpretations, conclusion and suggestions.

Step III: Provide recommendations after completing the entire process related to the model. It includes investment, resources, algorithms, techniques, etc.

2.5 Method of Analysis

The Simmer Package for Discrete- event simulation model was used in simulating the client's flow. Simmer is a process-oriented and trajectory-based Discrete-Event Simulation (DES) package for R designed to be a generic framework it leverages the power of RCPP to boost the performance and turning DES in R feasible. As a noteworthy characteristic, simmer exploits the concept of trajectory: a common path in the simulation model for entities of the same type. It is pretty simple to use and leverages the chaining/piping workflow introduced by the Magritt R package (Inaki et al., 2019c).

2.5.1 The Simmer Environment

First of all, we load the packages Simmer and Simmer.plot (Inaki et al., 2019b). Instantiate a new simulation environment, call it an ante-natal clinic, and create the client trajectory. Next, we define the processes trajectories and the completion time for the different activities as random draws from probability distributions or fixed variables (we are using fixed variables for this study since the model is deterministic). Likewise, the inter-arrival times for the clients are defined, see Appendix I.

The trajectory in Appendix I illustrates the two most basic activities available: displaying a message seize ( ) and spending some time in the system (timeout ( )). An arrival attached to this trajectory will execute the activities in the given order, i.e., it will display "Entering the trajectory", then it will spend some units of (simulated) time, and finally it will display "Leaving the trajectory". That is, the trajectory of an incoming client starts by seizing a nurse, it takes a fixed time of 15 minutes for a check-up and releases the nurse. Then, the client seizes the Doctor, takes a fixed time of 18 minutes for consultation and release the doctor. And at last, the client seizes the administration for 5 minutes

to schedule for the next appointment and release the administration. And the client leaves the system. Finally, the trajectory releases the client, so that it is ready again (Inaki et al., 2019a).

Once the processes trajectories are defined, the second block instantiates the simulation environment, we define the simulation time (start time and end time), creates resources (that is, append 5 identical nurses, 5 identical doctors and 3 identical administrators to the simulation environment), attaches a generator to the clients trajectory with inter-arrival time of 3 minutes and run the simulation for 480 units of time.

The simulation will be run for some units of times, and the simulator will monitor all the state changes and lifetimes of all processes, which enables any kind of analysis without any additional effort from the modeller's side (Inaki et al., 2019a).

Next, we use the visualization tools, that is the plot model with metric clients, the plot of resources with metrics the usage of a resource (nurse, doctor and administration) over the simulation time frame and the utilization of specified resources (nurse, doctor and administration) in the simulation, and the plot of arrivals with metrics activity time, waiting time and flow time.

Typically, running a certain simulation only once does not provide the information needed by the simulation analyst (Inaki, et al., 2019a). We will replicate the model execution, many times, with different initial conditions (by adding resources) and then perform another simulation over the output (Banks, 1991). Running simulation more than once can be achieved using the standard R tools, e.g. The lapply or simulator function. This study uses the lapply ( ) and Wrap ( ) to perform 100 replicates of the simulation, but the trajectory need not be redefined as shown in Appendix II.

3.0 DATA ANALYSIS

This chapter presents the results of Discrete-event simulation of clients' flow in Ante-natal clinic.

3.1 Presentation of Simulation results and Graph

Here, the number of servers is finite, so the clients are served according to a specific order, First-In-First-Out with a constant service time (the service consists of essentially the same routine task for all clients) and that the input parameters (service time) for the model used fixed numbers (single values) rather than a distribution in the function since the model is non-stochastic (hence deterministic). Also, the model assumes that all inter-arrival times equal some fixed constant. Therefore, the average inter-arrival time between clients is

$$\frac{(1.8+1.5+6)\,\text{min}\,utes}{3} = \frac{9.3\,\text{min}\,utes}{3} \approx 3\,\text{min}\,utes$$

At the client's trajectory

Nursing Station (Phase I): The client will seize one Nurse and remains for 15 minutes and releases the Nurse. Then moves from the Nursing station and joins the queue for consultation.

Doctors station (Phase II): The client will seize one Doctor for consultation and remains for 18 minutes and releases the Doctor. And then move to the administration.

Administration Station (Phase III): Here, the client will seize one administrator to schedule for the next appointment for at most 5 minutes and release the administration, and then exits the system as shown in the client's flow shown in Figure 1.

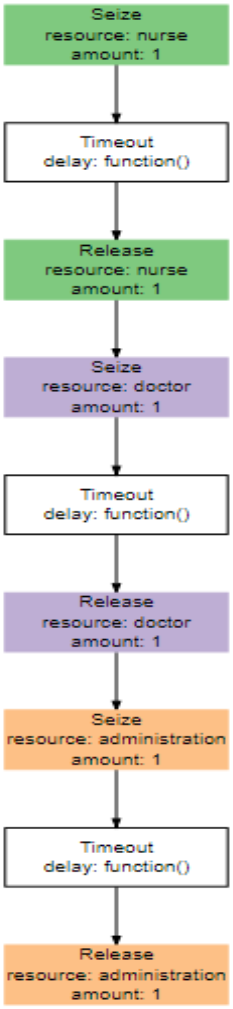

Figure 1 Clients flow plot

We add resources using the function add resources, we add Five Nurses, Five Doctors and Five Administration staff and add a generator. And simulate for 480 minutes (from 06:00 am to 02:00 pm) The simulation has been run for 100 times using the lapply function twice for the same empirical data obtained using the SIMMER package for Discrete-Event Simulation (Inputs and Outputs in Appendix Appendix III).

3.2 Simulation Results

3.2.1 First Simulation Output

Mean inter-arrival time = 3 minutes
Simulation for 0 to 8hours (i.e., 480minutes)
Number of clients generated = 160
Resource: Nurse
Server Status = 5(5)
Queue = 0 (Inf)
Resource: Doctor
Server Status = 5(5)
Queue = 25 (Inf)
Resource: Administration
Server Status = 1 (3)
Queue = 0 (Inf)
Simulation time = -7.992449 seconds

The metrics we are measuring from the resources are utilization and usage over time, and the metrics from arrivals are waiting time, activity time and flow time. Figures 2 to 6 will describe the metrics from resources and arrivals.

From the resource utilization plot in Figure 2, we can see that the Doctors and Nurses are 99% utilized which shows that their bottleneck at the Doctors and Nursing station.

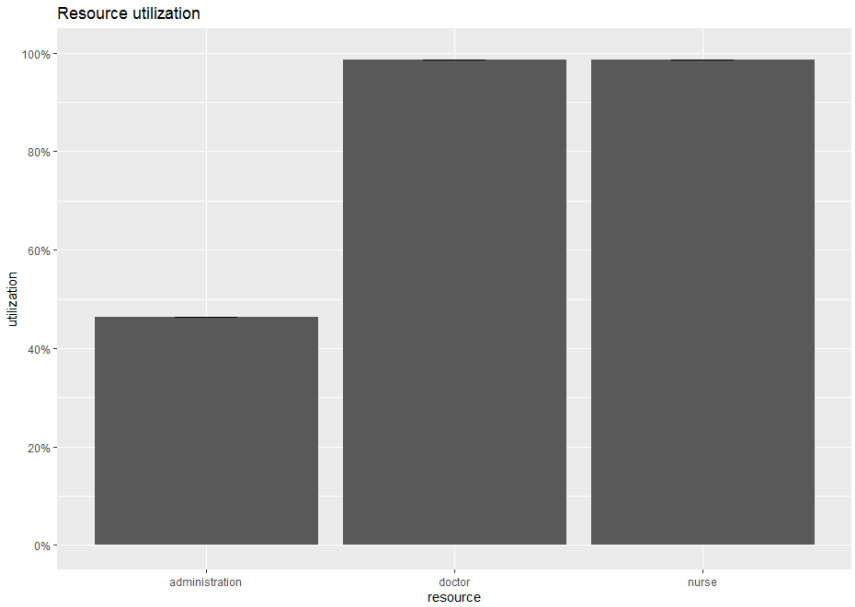

Figure 2 Resource Utilization Plot

From the resource usage plot in Figure 3, we can see in the graph field there are two lines the red line and the blue line. The red line represents the queue while the blue line represents the server. At the Doctors station, the red line (queue) crosses over the blue line (servers ceiling) which shows there is the bottleneck at the Doctors station. At the Nursing and Administration station (the queue line does not cross the server ceiling) which shows there is no bottleneck at the stations.

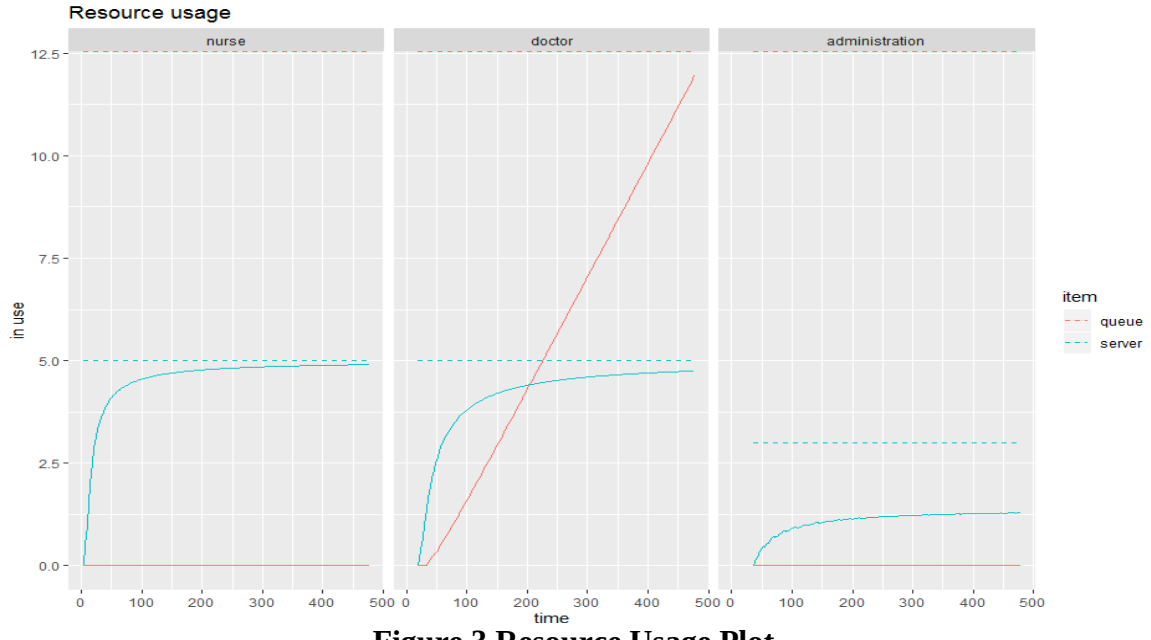

Figure 3 Resource Usage Plot

From the clients waiting time evolution plot shown in Figure 4, the clients waiting time is increasing by time. At the first 100 minutes, we can see that the waiting time was 11 minutes, as time goes on when the clients are filing up, the queue gets longer and the arrivals waiting time reaches up to 75 minutes. The blue line in the graph field is the trend line of 100 simulations and the black hair is the simulation of its own.

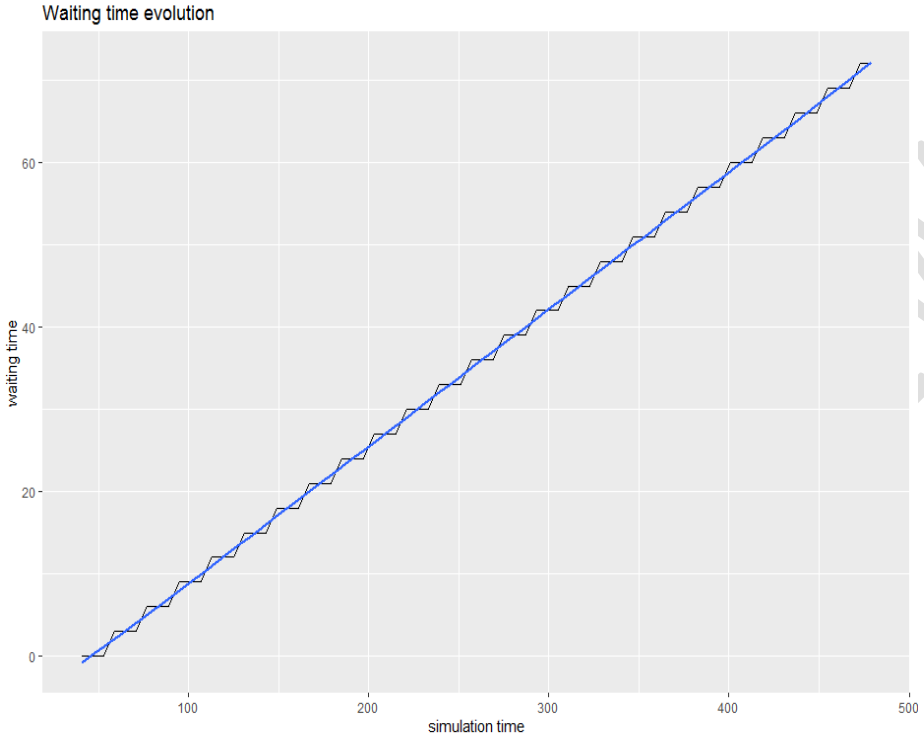

Figure 4 Waiting Time Evolution Plot

The activity time is constant over time (the service time is fixed). The blue line in Figure 5 represents the trend line.

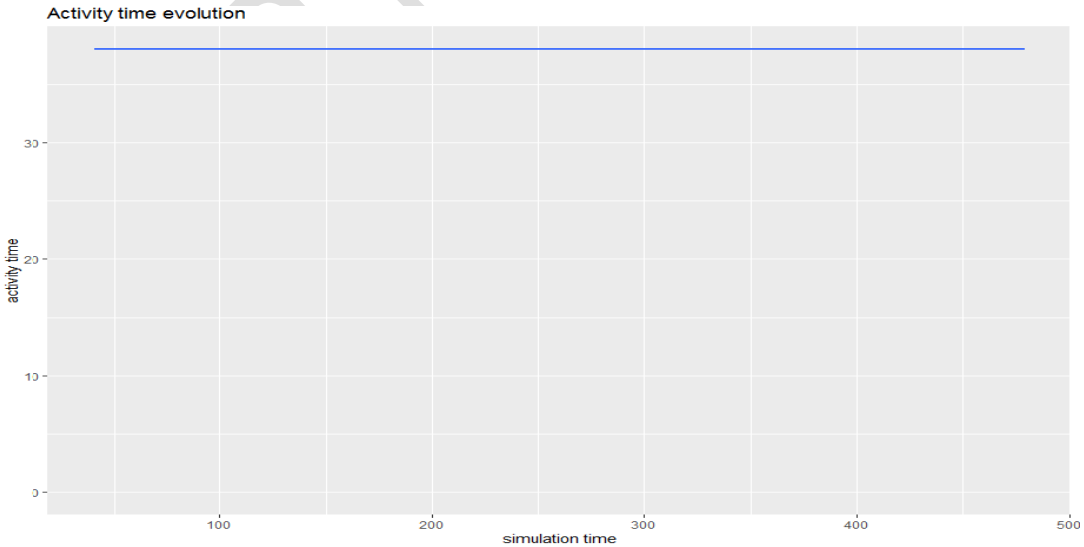

Figure 5 Activity Time Evolution

The flow time (total processing time) evolution plot in Figure 6 shows the time spent by clients in the system from beginning to end, as the time increases the flow time also increases, i.e. at 45 minutes the flow time was 38minutes, as the time goes up the clients processing time reaches to 95 minutes.

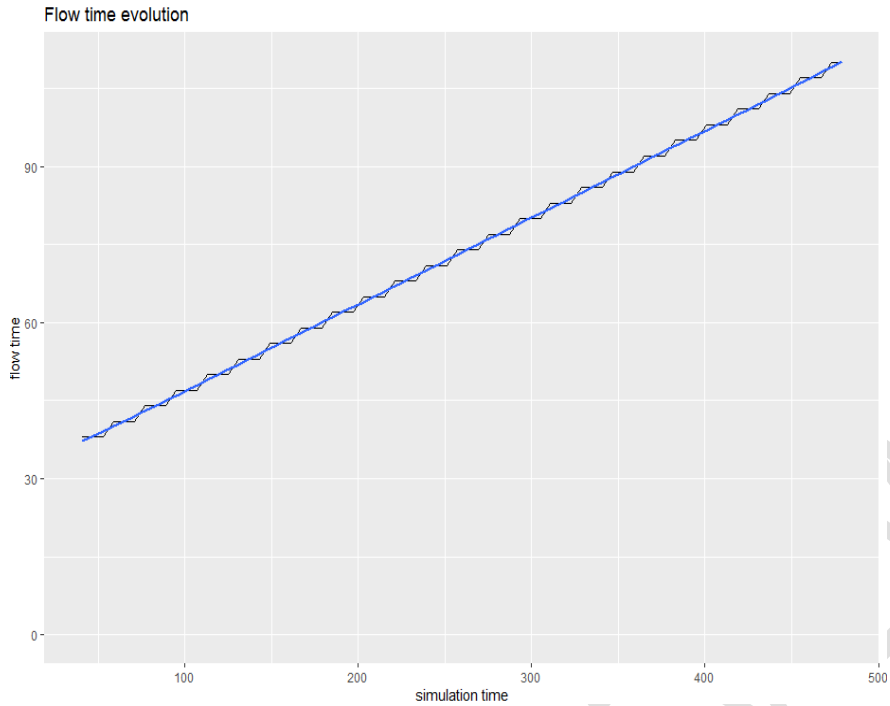

Figure 6 Flow Time Evolution

3.2.1 Second Simulation Outcome

We changed the settings from the Doctors station by adding one more Doctor and rerun the simulation.

Mean inter-arrival time = 3minutes
Simulation for 0 to 8hours (i.e., 480minutes)
Number of clients generated = 160
Resource: Nurse
Server Status = 5(5)
Queue = 0 (Inf)
Resource: Doctor
Server Status = 6(6)
Queue = 0 (Inf)
Resource: Administration
Server Status = 1 (3)
Queue = 0 (Inf)
Simulation time = -7.20seconds

The metrics we are measuring from the resources are utilization and usage over time, and the metrics from arrivals are waiting time, activity time and flow time. Figures 7 to 10b will describe the metrics from resources and arrivals.

From the resource utilization plot in Figure 7, the plot shows that the Doctors and Nurses are 96% utilized and there is no bottleneck at the Doctors station like in the previous simulation.

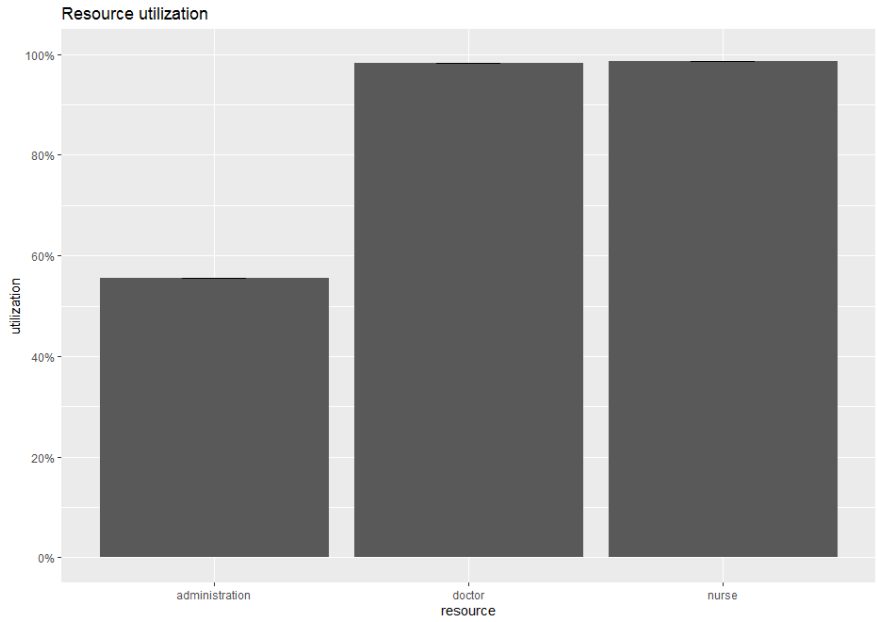

Figure. 7 Resource Utilization Plot

The resource usage plot shows that the queue at the Doctors station does not cross the server ceiling, which means that the bottleneck at the Doctors station disappeared by adding one Doctor. Also, there is no queue at the Nursing and Administration stations.

The resource usage plots in Figure 8 shows that the queue at the Doctors station does not cross the server ceiling, which means that the bottleneck at the Doctors station disappeared by adding one Doctor. Also, there is no queue at the Nursing and Administration stations.

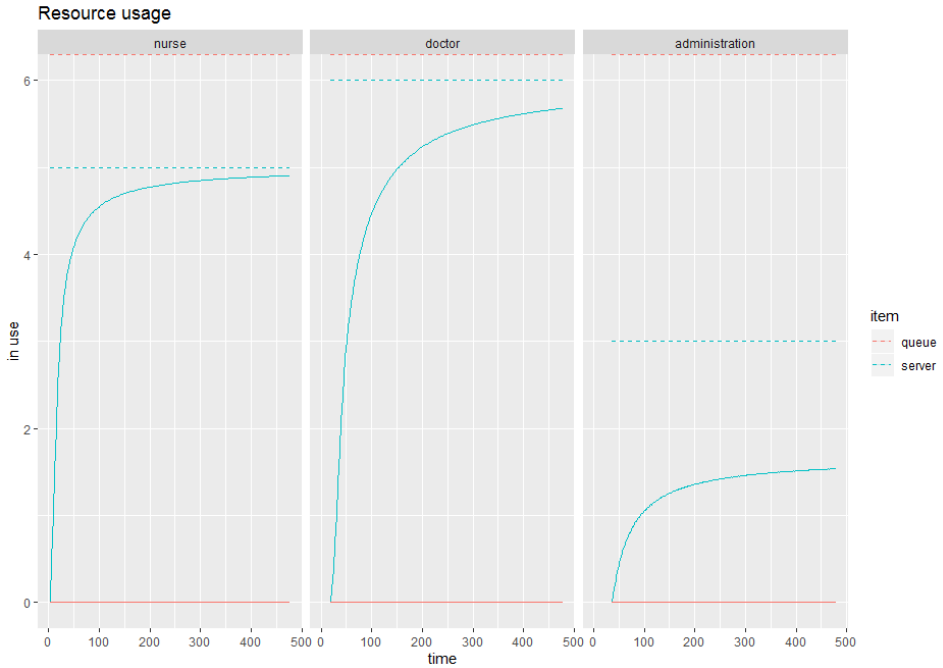

Figure. 8 Resource Usage Plot

By adding one more Doctor to the resources, as shown in Figure 9 in the graph the clients waiting time drops to zero.

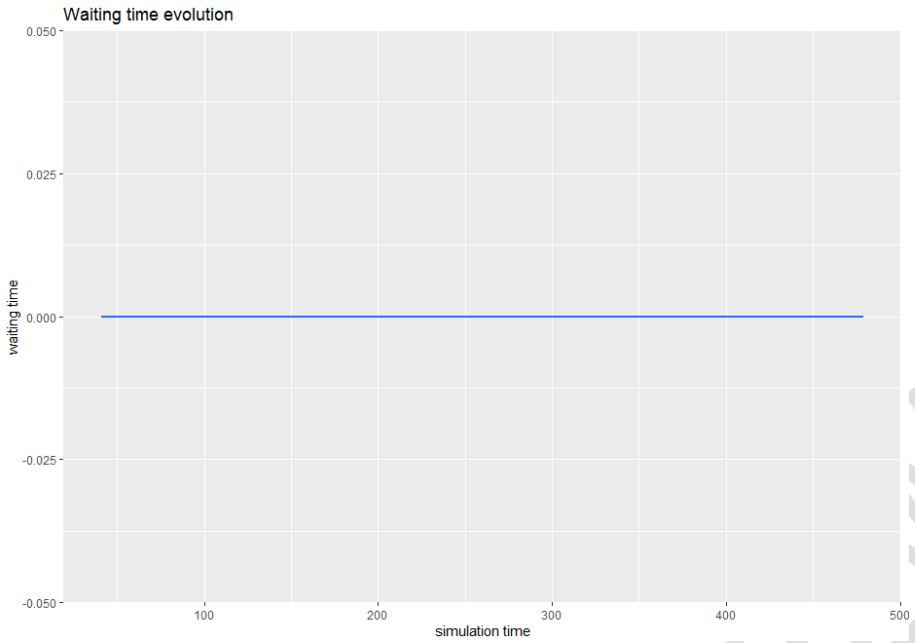

Figure. 9 Waiting Time Plot

The activity time and flow time are constant over time (the service time is fixed), as shown in Figure 10.

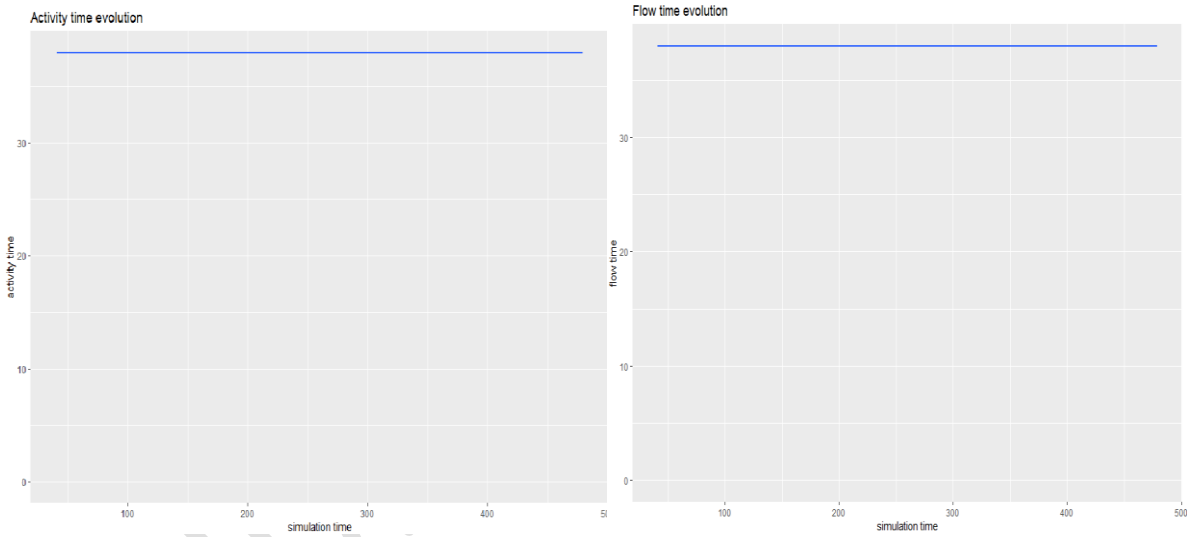

Figure 10a Activity Time Plot **Figure 10b Flow Time Plot**

4 DISCUSSION

The discrete event simulation model was used to analyze the system which measures the metrics (utilization and usage over time) of the resources (Doctors, Nurses and Administrators) and the metrics (waiting time, activity time, and flow time) of the clients. The simulation was replicated twice with 100 repetitions in each simulation for 480 minutes (8 hours).

Based on results obtained from the first simulation, with constant inter-arrival and service times, 160 clients were generated and 25clients waiting at the Doctors station. The graph of the server utilization shows that the Doctors and Nurses are highly utilized (there is the bottleneck at the Doctors and Nursing Stations). The resource usage plot shows the queue line is below then server ceiling at the Nurse and Administration station but crosses the server ceiling at the Doctors station which indicates

that from 0 to 8hours of the clinic services (0 – 480 minutes), the queue grows simultaneously at the Doctors station as the time of the day goes on.

The waiting time evolution plot shows at the first 100 minutes the waiting time was 11 minutes, and as time increases the waiting time of the clients also increases. The activity time is constant over time (service time) and from the total processing time (flow time) plot, the time clients spend in the system at 45 minutes of simulation was 38 minutes, but as the time increases the queue grows larger and the flow time increase to 110 minutes at 480 minutes simulation time.
We add one Doctor to the resources and replicate the simulation, the resource utilization plot shows that all the resources are highly utilized but there is no bottleneck at each station, in the resource usage plot it was discovered that the queue line at the Doctors station dropped (does not cross the server ceiling). The activity time is constant over time, the clients waiting time drops to zero and the clients flow time in the second simulation is constant (47 minutes for all clients).

The research concludes it is optimal to have six Doctors in the Ante-natal clinic. According to the study, the research has shown that Discrete-event Simulation Model is more accurate than other models to be used in simulating patients flow in hospitals and a tool for decision making on issues of resourcing and capacity planning.

5 CONCLUSION

Providing patients with timely access to appropriate medical care is an important element of health care delivery and increases patient satisfaction. "When" care is received, is often as important as "What" care is received. In this study, applications of queuing theory in modelling hospital services (antenatal clinic) has been used in measuring the performance of the resources (server utilization and usage) and clients flow time in the system since the healthcare facilities are interested in improving the system performance. The study establishes expecting mothers are not satisfied with long waiting times and experience negative effects as a result. Queuing simulation model is an effective and powerful modelling technique that can help the Hospital administrators to make decisions on staffing needs for optimal performance with regards to queuing challenges in the clinic. This study should, therefore, be replicated in other clinics and departments in Aminu Kano Teaching Hospital and other Hospitals as well to inform hospital administrators more on the usefulness of the application of queuing theory and simulation modelling as a tool for improved decision making with regards to the waiting line challenges that are faced by the hospital.

REFERENCES

Banks, Jerry (1991). Discrete-Event Simulation. In proceedings of the 1991 Winter Simulation Conference, pp. 1-224.

Hejase HJ, Hejase AJ, and Hejase HAN (2012). Quantitative Methods for Decision Makers: Management Approach. (2nd Edition), Philadelphia, USA: Masadir Inc.

Hejase AJ and Hejase HJ (2013). Research Methods: A Practical Approach for Business Students (2nd edition). Philadelphia, PA, USA: Masadir Inc.

Inaki U, and Bart S (2019a). **simmer**: Discrete-Event Simulation for R. R package version 4.3.0, URL https: //CRAN.R-project.org/package=simmer.

Inaki U, and Bart S (2019b). **simmer.plot**: Plotting methods for simmer. R package version 0.1.15, URL https: //CRAN.R-project.org/package=simmer.plot.

Inaki U, Bart S, and Arturo A, (2019c). Simmer: Discrete-Event Simulation for R, Universidad Carlos III de Madrid IMDEA Network Institute, pp. 1 – 21

Banks, Jerry (1999). Discrete-Event Simulation. In proceedings of the 1999 Winter Simulation Conference, pp. 7-13

Kipchumba Alfred (2016). Simulation of a Hospital Queuing System. International Journal of Engineering Modelling. pp. (4-5)

Shannon Robert E (1975). Systems Simulation: The Art and Science. New Jersey: Prentice Hall.

Law AM, and Kelton WD (2000). Simulation Modeling and Analysis, (3rd ed.). McGraw-Hill.

Green, L. 2006. Queuing Analysis in Healthcare. In Hall, R. (Ed.), Patient Flow: Reducing Delay in Healthcare Delivery (pp. 281-307). NY: Springer.

Jacobson H., Hall S and Swisher J (2006). Discreet-Event Simulation of Health Care Systems. In Hall, R. (Ed.), Patient Flow: Reducing Delay in Healthcare Delivery (pp. 210-252). Springer, NY

Mansharamani, Rajesh (1997). An overview of discrete event simulation methodologies and implementation. Sadhand, 22(5), pp. 611-627.

APPENDIX I

```
library(simmer)
library(simmer.plot)

# create environment
env <- simmer("antenatal clinic")
env


# create client trajectory
client <- trajectory (name = "client path", verbose = T)
client

## draw model
client %>%

  seize("nurse", 1) %>% ## need to define resources
  timeout(function() 15) %>%
  release("nurse", 1) %>%

  seize("doctor", 1) %>% ## need to define resources
  timeout(function() 18) %>%
  release("doctor", 1) %>%

  seize("administration", 1) %>% ## need to define resources
  timeout(function() 5) %>%
  release("administration", 1)
```

APPENDIX II

```
time1 = Sys.time()

envs <- lapply(1:100, function(i){
  simmer("antenatal clinic") %>%
    add_resource("nurse",5) %>%
    add_resource("doctor",5) %>%
    add_resource("administration", 3) %>%
    add_generator(name_prefix = "sim_client_",
                  trajectory = client,
                  distribution = function() 3)%>%

    run(480) %>%
    wrap()
})

time2 = Sys.time()
time1 - time2

#plot the model
plot(client)
plot(client, verbose = T)

resources <- get_mon_resources(envs)
plot(resources, metric = "utilization")

plot(resources, metric = "usage", c("nurse", "doctor", "administration"),
                          items = c ("queue", "server"))


arrivals <- get_mon_arrivals(envs)
plot(arrivals, metric = "waiting_time")
plot(arrivals, metric = "activity_time")
plot(arrivals, metric = "flow_time")
```

APPENDIX III

```
library(simmer)
library(simmer.plot)

# create environment
env <- simmer("antenatal clinic")
env


# create client trajectory
client <- trajectory (name = "client path", verbose = T)
client

## draw model
client %>%

  seize("nurse", 1) %>% ## need to define resources
  timeout(function() 15) %>%
  release("nurse", 1) %>%

  seize("doctor", 1) %>% ## need to define resources
  timeout(function() 18) %>%
  release("doctor", 1) %>%

  seize("administration", 1) %>% ## need to define resources
  timeout(function() 5) %>%
  release("administration", 1)

time1 = Sys.time()

envs <- lapply(1:100, function(i){
  simmer("antenatal clinic") %>%
    add_resource("nurse",5) %>%
    add_resource("doctor",5) %>%
    add_resource("administration", 3) %>%
    add_generator(name_prefix = "sim_client_",
                  trajectory = client,
                  distribution = function() 3)%>%

    run(480) %>%
    wrap()
})

time2 = Sys.time()
time1 - time2

#plot the model
plot(client)
plot(client, verbose = T)

resources <- get_mon_resources(envs)
plot(resources, metric = "utilization")

plot(resources, metric = "usage", c("nurse", "doctor", "administration"),
                          items = c ("queue", "server"))


arrivals <- get_mon_arrivals(envs)
plot(arrivals, metric = "waiting_time")
plot(arrivals, metric = "activity_time")
plot(arrivals, metric = "flow_time")
```

Output Result:

```
Loading required package: poisson
> library(simmer)
> library(simmer.plot)
Loading required package: ggplot2

Attaching package: 'simmer.plot'

The following objects are masked from 'package:simmer':

    get_mon_arrivals, get_mon_attributes, get_mon_resources

> # create environment
> env <- simmer("antenatal clinic")
> env
simmer environment: antenatal clinic | now: 0 | next:
{ Monitor: in memory }
> # create client trajectory
> client <- trajectory (name = "client path", verbose = T)
> client
trajectory: client path, 0 activities
> ## draw model
> client %>%
+
+   seize("nurse", 1) %>% ## need to define resources
+   timeout(function() 15) %>%
+   release("nurse", 1) %>%
+
+   seize("doctor", 1) %>% ## need to define resources
+   timeout(function() 18) %>%
+   release("doctor", 1) %>%
+
+   seize("administration", 1) %>% ## need to define resources
+   timeout(function() 5) %>%
+   release("administration", 1)
trajectory: client path, 9 activities
{ Activity: Seize        |         0 <- 0x7407c2ee50 -> 0x74078f4900 | resource: nurse, amount: 1 }
{ Activity: Timeout      | 0x7407c2ee50 <- 0x74078f4900 -> 0x7407c227a0 | delay: function() }
{ Activity: Release      | 0x74078f4900 <- 0x7407c227a0 -> 0x7401263a50 | resource: nurse, amount: 1
 }
{ Activity: Seize        | 0x7407c227a0 <- 0x7401263a50 -> 0x74078f4b80 | resource: doctor, amount:
 1 }
{ Activity: Timeout      | 0x7401263a50 <- 0x74078f4b80 -> 0x7407c22860 | delay: function() }
{ Activity: Release      | 0x74078f4b80 <- 0x7407c22860 -> 0x7401263b20 | resource: doctor, amount:
 1 }
{ Activity: Seize        | 0x7407c22860 <- 0x7401263b20 -> 0x74078f4bc0 | resource: administration,
 amount: 1 }
{ Activity: Timeout      | 0x7401263b20 <- 0x74078f4bc0 -> 0x7407c22440 | delay: function() }
{ Activity: Release      | 0x74078f4bc0 <- 0x7407c22440 -> 0         | resource: administration, amo
unt: 1 }
```

```
> envs <- lapply(1:100, function(i){
+   simmer("antenatal clinic") %>%
+     add_resource("nurse",5) %>%
+     add_resource("doctor",5) %>%
+     add_resource("administration", 3) %>%
+     add_generator(name_prefix = "sim_client_",
+                   trajectory = client,
+                   distribution = function() 3)%>%
+
+     run(480) %>%
+     wrap()
+ })
>   simmer("antenatal clinic") %>%
+     add_resource("nurse",5) %>%
+     add_resource("doctor",5) %>%
+     add_resource("administration", 3) %>%
+     add_generator(name_prefix = "sim_client_",
+                   trajectory = client,
+                   distribution = function() 3)%>%
+
+     run(480) %>%
+     wrap()
simmer environment: antenatal clinic | now: 480 | next: 480
{ Monitor:  }
{ Resource: nurse | monitored: TRUE | server status: 5(5) | queue status: 0(Inf) }
{ Resource: doctor | monitored: TRUE | server status: 5(5) | queue status: 25(Inf) }
{ Resource: administration | monitored: TRUE | server status: 1(3) | queue status: 0(Inf) }
{ Source: sim_client_ | monitored: 1 | n_generated: 160 }
```

431

```
> time1 = Sys.time()
>
> envs <- lapply(1:100, function(i){
+   simmer("antenatal clinic") %>%
+     add_resource("nurse",5) %>%
+     add_resource("doctor",5) %>%
+     add_resource("administration", 3) %>%
+     add_generator(name_prefix = "sim_client_",
+                   trajectory = client,
+                   distribution = function() 3)%>%
+
+     run(480) %>%
+     wrap()
+ })
>
> time2 = Sys.time()
> time1 - time2
Time difference of -7.375293 secs
> #plot the model
> plot(client)
> plot(client, verbose = T)
> resources <- get_mon_resources(envs)
> plot(resources, metric = "utilization")
> plot(resources, metric = "usage", c("nurse", "doctor", "administration"),
+                           items = c ("queue", "server"))
> arrivals <- get_mon_arrivals(envs)
> plot(arrivals, metric = "waiting_time")
`geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
> plot(arrivals, metric = "activity_time")
`geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
> plot(arrivals, metric = "flow_time")
`geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```

432