Case Study 5

Jonathan Busch

2024-04-28

First we read in the given data to determine the proportion of types of patients and the distribution of arrival times.

library(readr)  
Case5\_emergency\_room <- read\_csv("C:/Users/jj\_bu/Desktop/TTU/Classes/Spring/Simulation and Optimization/Case Study/Question Five/Case5\_emergency-room.csv")

## Rows: 100 Columns: 2  
## ── Column specification ────────────────────────────────────────────────────────  
## Delimiter: ","  
## chr (1): type  
## dbl (1): interArrival  
##   
## ℹ Use `spec()` to retrieve the full column specification for this data.  
## ℹ Specify the column types or set `show\_col\_types = FALSE` to quiet this message.

#View(Case5\_emergency\_room)

Cullen and Frey chart and fitting distributions. Due to the nature of the data we will fit to continous distributions.

library(fitdistrplus)

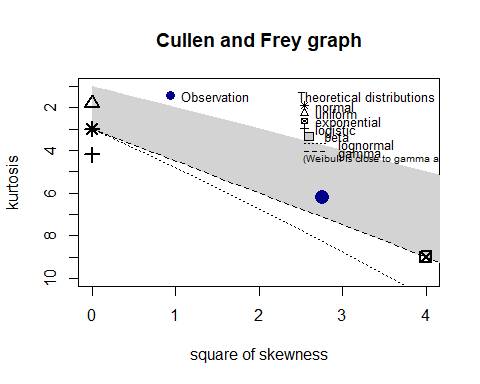
## Warning: package 'fitdistrplus' was built under R version 4.2.3

## Loading required package: MASS

## Warning: package 'MASS' was built under R version 4.2.3

## Loading required package: survival

descdist(Case5\_emergency\_room$interArrival,discrete=FALSE)



## summary statistics  
## ------  
## min: 0.1 max: 72.4   
## median: 10.2   
## mean: 15.077   
## estimated sd: 14.35904   
## estimated skewness: 1.661798   
## estimated kurtosis: 6.224021

This graph shows we should check the lognormal, gamma, and weibull distributions.

Now we will fit the data to these distributions.

fit.weibull=fitdist(Case5\_emergency\_room$interArrival,"weibull")  
summary(fit.weibull)

## Fitting of the distribution ' weibull ' by maximum likelihood   
## Parameters :   
## estimate Std. Error  
## shape 0.9979736 0.07955256  
## scale 15.0680617 1.58504965  
## Loglikelihood: -371.3167 AIC: 746.6335 BIC: 751.8438   
## Correlation matrix:  
## shape scale  
## shape 1.0000000 0.3036705  
## scale 0.3036705 1.0000000

fit.lognormal=fitdist(Case5\_emergency\_room$interArrival,"lnorm")  
summary(fit.lognormal)

## Fitting of the distribution ' lnorm ' by maximum likelihood   
## Parameters :   
## estimate Std. Error  
## meanlog 2.111460 0.13654252  
## sdlog 1.365425 0.09654991  
## Loglikelihood: -384.1864 AIC: 772.3729 BIC: 777.5832   
## Correlation matrix:  
## meanlog sdlog  
## meanlog 1 0  
## sdlog 0 1

fit.gamma=fitdist(Case5\_emergency\_room$interArrival,"gamma")  
summary(fit.gamma)

## Fitting of the distribution ' gamma ' by maximum likelihood   
## Parameters :   
## estimate Std. Error  
## shape 0.96354630 0.11955644  
## rate 0.06388923 0.01025346  
## Loglikelihood: -371.2718 AIC: 746.5435 BIC: 751.7539   
## Correlation matrix:  
## shape rate  
## shape 1.0000000 0.7728234  
## rate 0.7728234 1.0000000

These results show that the gamma distribution is the best fit due to its highest log liklihood and lowest AIC and BIC.

Now we will perform a chi squared test to see if we fail to reject this fitted distrubtion.

g=gofstat(fit.gamma)  
g$chisqpvalue

## [1] 0.1704838

With a p-value of 0.17 we fail to reject the null hypothesis that the fit is gamma. Therefore we will use an arrival distribution of gamma with a shape of 0.964 and a rate of 0.064

Now we will check the proportions of patient types.

mean(Case5\_emergency\_room$type=='NIA')

## [1] 0.18

mean(Case5\_emergency\_room$type=='CW')

## [1] 0.82

length(Case5\_emergency\_room$type)

## [1] 100

The proportion of patients that need immediate attention is 0.18% and the proportion of patients that can wait is 0.82%. We will use these as the estimated proportions for building our simulation.

Now we will build the simulation of the emergency room.

library(simmer)

## Warning: package 'simmer' was built under R version 4.2.3

##   
## Attaching package: 'simmer'

## The following object is masked from 'package:MASS':  
##   
## select

set.seed(123)  
  
emergencyRoom=trajectory("ER")%>%  
 branch(option=function() sample(1:2,1,prob = c(.18,.82),replace=T),continue=c(T,T),  
 trajectory("High Priority")%>%  
 set\_attribute("Priority",3)%>%  
 set\_prioritization(c(3,7,T))%>%  
 seize('doctor')%>%  
 timeout(function() runif(1,10,70))%>%  
 release('doctor')%>%  
 set\_prioritization(c(2,7,T))%>%  
 seize("doctor")%>%  
 timeout(function() runif(1,10,50))%>%  
 release('doctor'),  
 trajectory("Low Priority")%>%  
 set\_attribute("Priority",1)%>%  
 set\_prioritization(c(1,7,T))%>%  
 seize('doctor')%>%  
 timeout(function() runif(1,5,25))%>%  
 release("doctor")%>%  
 set\_prioritization(c(2,7,T))%>%  
 seize('doctor')%>%  
 timeout(function() runif(1,5,15))%>%  
 release('doctor')  
 )  
  
envs=lapply(1:20, function(i){  
 simmer("ER")%>%  
 add\_resource("doctor",2)%>%  
 add\_generator("patient",emergencyRoom,function() rgamma(1,shape=0.964,rate=0.064),mon = 2)%>%  
 run(1440)  
   
   
   
   
   
   
})

Now that we have a simulated model we can inspect where possible bottlenecks are and figure out to improve the system.

library(simmer.plot)

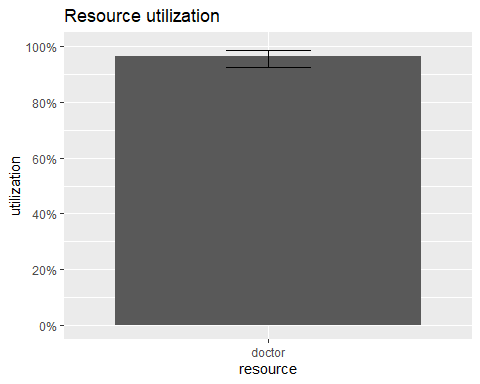
## Warning: package 'simmer.plot' was built under R version 4.2.3

## Loading required package: ggplot2

##   
## Attaching package: 'simmer.plot'

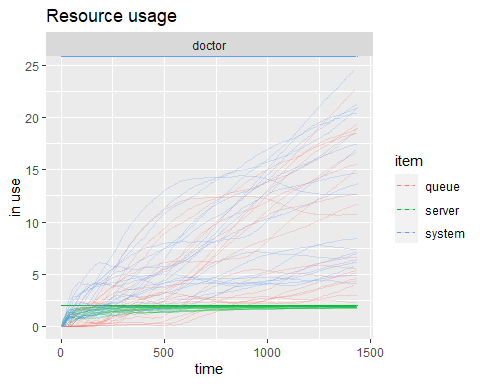
## The following objects are masked from 'package:simmer':  
##   
## get\_mon\_arrivals, get\_mon\_attributes, get\_mon\_resources

plot(get\_mon\_resources(envs),metric="utilization")



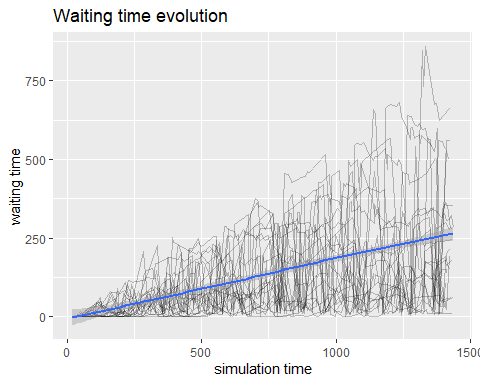
From this we can see that the doctors are hitting near maximum utilization. This could imply that we would need to add extra doctors as the time in which max utilization does not occur is most likely when the day has just began and the queue has not grown.

plot(get\_mon\_resources(envs),metric="usage")



plot(get\_mon\_arrivals(envs),metric="waiting\_time")

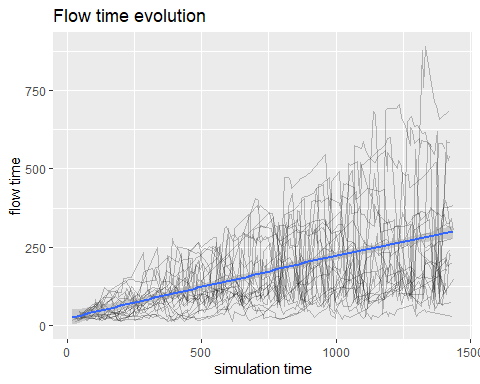
## `geom\_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'



This graph shows that our waiting time is constantly growing meaning the current system is not sufficient in keeping the queue from continuously growing.

plot(get\_mon\_arrivals(envs),metric="flow\_time")

## `geom\_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'



Similar to the waiting time, our flow time continuously grows throughout the day. Now we will compare the flows times of the two types of patients to see if there is any major hold ups between the two types of patients.

x1=get\_mon\_arrivals(envs)  
x2=get\_mon\_attributes(envs)  
  
all=merge(x1,x2,by=c("name","replication"),all=TRUE)  
all=na.omit(all)  
  
High=subset(all,all$value==3)  
Low=subset(all,all$value==1)  
  
High.flowTime=(High$end\_time-High$start\_time)  
Low.flowTime=(Low$end\_time-Low$start\_time)  
  
cat("The average of the flow time for high priority patients is:",mean(High.flowTime),'\n')

## The average of the flow time for high priority patients is: 108.8853

cat("The average of the flow time for low priority patients is:",mean(Low.flowTime),'\n')

## The average of the flow time for low priority patients is: 185.2457

From these results we can see that low priority patients have a much higher flow time than low priority patients. We can also see that the time to go through the doctor visits is quite long for both types of patients but the low priority is taking almost twice as long as the high priority.

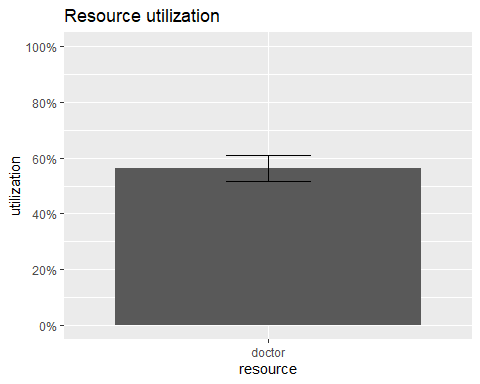
We will try three different tactics in order to see if they improve our flow times. First we will try adding extra doctors to see if additional hires will improve the system. Then we will try a system that splits the queue into two with the low and high priority patients being placed into two different queues.

First up is simply adding more doctors. In this case we will test 4 different doctors.

set.seed(123)  
  
emergencyRoom=trajectory("ER")%>%  
 branch(option=function() sample(1:2,1,prob = c(.18,.82),replace=T),continue=c(T,T),  
 trajectory("High Priority")%>%  
 set\_attribute("Priority",3)%>%  
 set\_prioritization(c(3,7,T))%>%  
 seize('doctor')%>%  
 timeout(function() runif(1,10,70))%>%  
 release('doctor')%>%  
 set\_prioritization(c(2,7,T))%>%  
 seize("doctor")%>%  
 timeout(function() runif(1,10,50))%>%  
 release('doctor'),  
 trajectory("Low Priority")%>%  
 set\_attribute("Priority",1)%>%  
 set\_prioritization(c(1,7,T))%>%  
 seize('doctor')%>%  
 timeout(function() runif(1,5,25))%>%  
 release("doctor")%>%  
 set\_prioritization(c(2,7,T))%>%  
 seize('doctor')%>%  
 timeout(function() runif(1,5,15))%>%  
 release('doctor')  
 )  
  
envs=lapply(1:20, function(i){  
 simmer("ER")%>%  
 add\_resource("doctor",4)%>%  
 add\_generator("patient",emergencyRoom,function() rgamma(1,shape=0.964,rate=0.064),mon = 2)%>%  
 run(1440)  
   
   
   
   
   
   
})

First lets take a look at the resource utilization to see if there is any change

plot(get\_mon\_resources(envs),metric="utilization")



We see massive changes to the doctor utilization providing evidence that additional hires will have positive effects on the flow time.

Now we will see the flow times of each group to see if there is any improvement.

x1=get\_mon\_arrivals(envs)  
x2=get\_mon\_attributes(envs)  
  
all=merge(x1,x2,by=c("name","replication"),all=TRUE)  
all=na.omit(all)  
  
High=subset(all,all$value==3)  
Low=subset(all,all$value==1)  
  
High.flowTime=(High$end\_time-High$start\_time)  
Low.flowTime=(Low$end\_time-Low$start\_time)  
  
cat("The average of the flow time for high priority patients is:",mean(High.flowTime),'\n')

## The average of the flow time for high priority patients is: 75.13857

cat("The average of the flow time for low priority patients is:",mean(Low.flowTime),'\n')

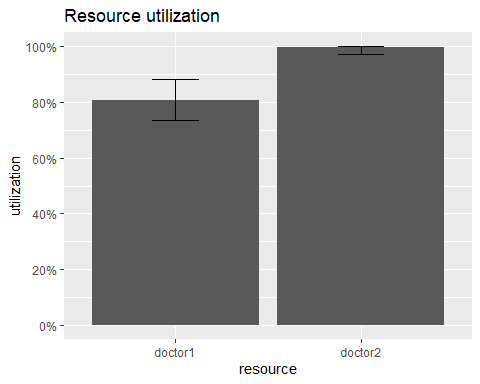
## The average of the flow time for low priority patients is: 28.0742

Adding two extra doctors had significantly reduced the flow time for both types of patients. It appears that high priority patients are taking significantly more time to go through the ER.

Now that we have found that additional doctors will lead to improvements let us now see how if changing the nature of the queues will lead to improvements in the queue time. The main suggestion here is to have a separate doctor and queue for each type of patient.

set.seed(123)  
  
emergencyRoom=trajectory("ER")%>%  
 branch(option=function() sample(1:2,1,prob = c(.18,.82),replace=T),continue=c(T,T),  
 trajectory("High Priority")%>%  
 set\_attribute("Priority",3)%>%  
 seize('doctor1')%>%  
 timeout(function() runif(1,10,70))%>%  
 release('doctor1')%>%  
 seize("doctor1")%>%  
 timeout(function() runif(1,10,50))%>%  
 release('doctor1'),  
 trajectory("Low Priority")%>%  
 set\_attribute("Priority",1)%>%  
 seize('doctor2')%>%  
 timeout(function() runif(1,5,25))%>%  
 release("doctor2")%>%  
 seize('doctor2')%>%  
 timeout(function() runif(1,5,15))%>%  
 release('doctor2')  
 )  
  
envs=lapply(1:20, function(i){  
 simmer("ER")%>%  
 add\_resource("doctor1",1)%>%  
 add\_resource("doctor2",1)%>%  
 add\_generator("patient",emergencyRoom,function() rgamma(1,shape=0.964,rate=0.064),mon = 2)%>%  
 run(1440)  
   
   
   
   
   
   
})

plot(get\_mon\_resources(envs),metric="utilization")



We can see from having separate queues with a doctor assigned to each doctor 1 is not utilized as much as doctor 2 perhaps showing that doctor one is idling somewhat creating an inefficiency.

Now we can check the flow times of each type of patient once more.

x1=get\_mon\_arrivals(envs)  
x2=get\_mon\_attributes(envs)  
  
all=merge(x1,x2,by=c("name","replication"),all=TRUE)  
all=na.omit(all)  
  
High=subset(all,all$value==3)  
Low=subset(all,all$value==1)  
  
High.flowTime=(High$end\_time-High$start\_time)  
Low.flowTime=(Low$end\_time-Low$start\_time)  
  
cat("The average of the flow time for high priority patients is:",mean(High.flowTime),'\n')

## The average of the flow time for high priority patients is: 188.7886

cat("The average of the flow time for low priority patients is:",mean(Low.flowTime),'\n')

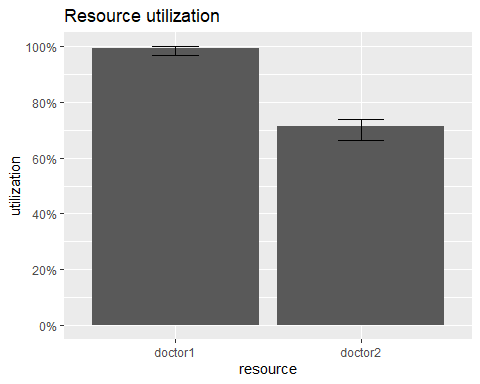
## The average of the flow time for low priority patients is: 312.3444

As expected from the utilization graph this system is much worse than the original.

Now we will try a system in which one doctor is assigned to see all of the patients first treatment and another doctor to see all of the patients second treatment. We will leave the priorities intact as this is likely an important constraint in the system. We do not want to ignore patients that are bleeding out for someone with a cold.

set.seed(123)  
  
emergencyRoom=trajectory("ER")%>%  
 branch(option=function() sample(1:2,1,prob = c(.18,.82),replace=T),continue=c(T,T),  
 trajectory("High Priority")%>%  
 set\_attribute("Priority",3)%>%  
 set\_prioritization(c(3,7,T))%>%  
 seize('doctor1')%>%  
 timeout(function() runif(1,10,70))%>%  
 release('doctor1')%>%  
 set\_prioritization(c(2,7,T))%>%  
 seize("doctor2")%>%  
 timeout(function() runif(1,10,50))%>%  
 release('doctor2'),  
 trajectory("Low Priority")%>%  
 set\_attribute("Priority",1)%>%  
 set\_prioritization(c(1,7,T))%>%  
 seize('doctor1')%>%  
 timeout(function() runif(1,5,25))%>%  
 release("doctor1")%>%  
 set\_prioritization(c(2,7,T))%>%  
 seize('doctor2')%>%  
 timeout(function() runif(1,5,15))%>%  
 release('doctor2')  
 )  
  
envs=lapply(1:20, function(i){  
 simmer("ER")%>%  
 add\_resource("doctor1",1)%>%  
 add\_resource("doctor2",1)%>%  
 add\_generator("patient",emergencyRoom,function() rgamma(1,shape=0.964,rate=0.064),mon = 2)%>%  
 run(1440)  
   
   
   
   
   
   
})

plot(get\_mon\_resources(envs),metric="utilization")



x1=get\_mon\_arrivals(envs)  
x2=get\_mon\_attributes(envs)  
  
all=merge(x1,x2,by=c("name","replication"),all=TRUE)  
all=na.omit(all)  
  
High=subset(all,all$value==3)  
Low=subset(all,all$value==1)  
  
High.flowTime=(High$end\_time-High$start\_time)  
Low.flowTime=(Low$end\_time-Low$start\_time)  
  
cat("The average of the flow time for high priority patients is:",mean(High.flowTime),'\n')

## The average of the flow time for high priority patients is: 96.98072

cat("The average of the flow time for low priority patients is:",mean(Low.flowTime),'\n')

## The average of the flow time for low priority patients is: 239.0166

These results suggest that this change leads to a smaller flow time for our high priority patients but the low priority patients must wait longer in this system. Therefore, assigning one doctor as a first treatment doctor and another as a final treatment doctor will make lead to quicker flow time for high priority patients and slower times for those with less severe conditions. In certain situations this may be desirable to get high priority patients through the system quicker.

Now the questions will be answered to serve as a summary of the findings.

1. Analyze your results and explain your suggestions for reducing the waiting time of the patients.

Our results indicated that the resource utilization of the doctors was fairly high in the original simulation. Therefore I proposed 3 different changes in order to see if they make improvements on the system. The first proposal was to simply hire additional doctors to see if this would make changes to the average flow time of the patients. The second was to assign each doctor an individual queue made up of each type of patient. The last model was to assign one doctor for the first treatment of patients and another doctor to the final treatment of patients.

1. What is the average flow-time for NIA and CW patients before or after applying different suggestions

Original Model

| Priority | Flow Time |
| --- | --- |
| High | 108.89 |
| Low | 185.25 |

Original System with 2 Additional Doctors (Four total)

| Priority | Flow Time |
| --- | --- |
| High | 75.14 |
| Low | 28.07 |

Doctors Assigned to own queue for each patient type

| Priority | Flow Time |
| --- | --- |
| High | 188.79 |
| Low | 312.34 |

Doctors assigned to either first treatment or final treatment

| Priority | Flow Time |
| --- | --- |
| High | 96.98 |
| Low | 239.02 |

1. Discuss the utilization of doctors before or after applying suggestions.

In the original model both doctors are utilized nearly to there fullest. This indicates that they are being used efficiently as there is little to no idle time for the doctors. If a doctor is not currently being utilized but a queue still remains, we can consider the setup to be inefficient as that idle doctor could be contributing towards lowering the queue.

In the original system but with 2 additional doctors we see that utilization drops to nearly 60% for all four doctors. While our flow times become dramatically smaller, the utilization indicates we may actually only need to hire one additional doctor to see keep up with demand for treatment rather than two.

In the doctors assigned to their own queue for each patient type we see a utilization of doctor one(high priority doctor) around 80% and a utilization of doctor 2 around 100%(low priority doctor). We also see larger queue times especially for the low priority patients. This indicates that the biggest use of resources are the low priority patients as doctor 2 is swamped dealing with this group while doctor 1 has some idle time. This system could work if we had more doctors and weighted more of these doctors to fulfilling the low priority patients role.

Finally in the doctors assigned to either first or final treatment we see doctor 1 (the first treatment doctor) having a utilization of near 100% and doctor 2 (final treatment doctor) with a utilization of nearly 70%. This indicates that the majority of the work occurring is in the initial treatment. We do however see flow times more comparable with the original setup. A setup like this could also function if additional hired doctors were weighted more towards fulfilling initial treatments. Furthermore, this setup may be desirable if we wish to minimize the waiting time for high priority patients at the expense for waiting time for low priority patients.