Simulation Supplement

James Muguira

April 25, 2016

# Synopsis

Simulation is a useful way to understand how to optimize a system. Many times, simulation is applied to understand systems that are too dangerous to create and test in real-life. Simulation as an optimization technique is a part of the larger class of Operations Research problems.

This study will explore the problem of customers waiting in line for a sandwich. This problem falls into the general class of a single server queue and is referred to as an M/M/1 queuing theory problem. The M/M/1 comes from the definition of the system as composed of an arrival process, which is Markovian (M), a service time distribution (M), which is again Markovian, and a single server. The arrival time distribution, for our purposes, is characterized as a Poisson process. The server time distribution could be Poisson or exponential.

The Poisson distribution is a good distribution to employ in this case because each customer arriving at the checkout is independent of other customers. Observe that more customers enter the shop during lunch break. This implies that the customers are not exactly independent over the course of a day. However, if we constrain our model to just the lunch hour (e.g. from 11-1) we still obey the constraints of the Poisson distribution.

# Methods

This exercise will use the *simmer* R package, which can be installed from CRAN. Simmer is a type of simulator known as a Discrete Event Simulator (DES) (see the lecture notes for other types). It designed to be a framework for building and exploring simulation.

This example will consider a sandwich shop during the hours of 11 AM to 1 PM. We will explore the business problem of utilization of servers and from the customer side: length of wait for a sandwich.

To setup for using simmer we have to create trajectory that defines our system. In this case, we'll have:

* A sandwich\_maker that can serve 1 customer at a time.
* The sandwich\_maker takes between 1 and 5 minutes to make a sandwich (i.e. rnorm(1, 5)).
* The sandwich process includes a few minutes wait as a sandwich warms in an oven.

The terminology here is that a sandwich\_maker will seize 1 customer from the input line and timeout or be busy for between 1 and 5 minutes and then release that single customer (i.e. the customer pays and leaves).

library(simmer)

## Warning: package 'simmer' was built under R version 3.2.5

t0 <- create\_trajectory("sandwich shop") %>%  
 seize("sandwich\_maker", 1) %>%  
 timeout(function() rnorm(1, 5)) %>%  
 release("sandwich\_maker", 1)

To run this simulation 1 time issue the follow command. Simmer tells you it created a sandwich\_maker and a customer. The run(until=120) means we run the simulation for 120 minutes.

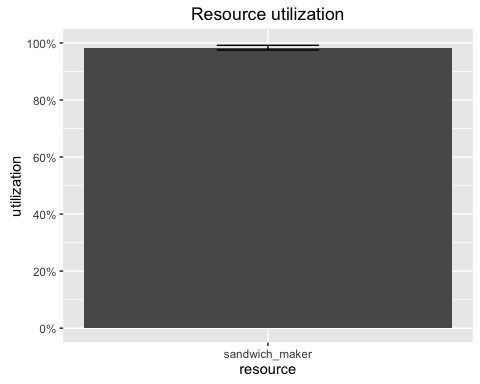
env <- simmer("Sandwich Joe") %>%  
 add\_resource("sandwich\_maker", 1) %>%  
 add\_generator("customer", t0, function() rnorm(1, 10, 1))  
#  
 env %>% run(until=120)

## simmer environment: Sandwich Joe | now: 122.343604524504 | next: 122.343604524504  
## { Resource: sandwich\_maker | monitored: 1 | server status: 0(1) | queue status: 0(Inf) }  
## { Generator: customer | monitored: 1 | n\_generated: 13 }

* The simmer add\_resource command adds a single service resource named "sandwich\_maker" into the simulation system.
* The simmer add\_generator command adds a single customer to the simulation (referred to as t0). The arrival time is Poisson distributed between 1 and 10 minutes.

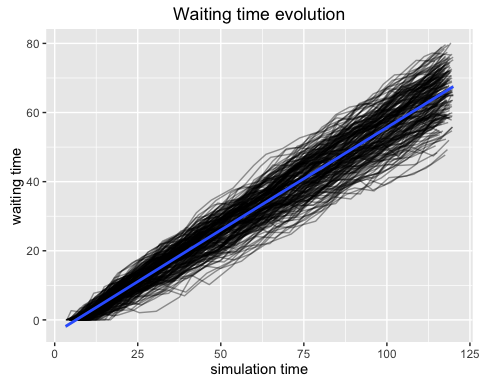
When you use simulation to optimize you NEVER run things once. You run it a bunch of times and take the average. So, let's run the above simulation (called env) 200 times and see how busy the sandwich\_maker is:

env <- lapply(1:200, function(i) {  
 simmer("Sandwich Joe") %>%  
 add\_resource("sandwich\_maker", 1) %>%  
 add\_generator("customer", t0, function() rpois(1, 2)) %>%  
 # limit run to 120 time units!  
 run(120)  
})  
plot\_resource\_utilization(env, c("sandwich\_maker"))



Ok, the sandwich\_maker seems pretty busy at lunch time. How long does a customer wait?

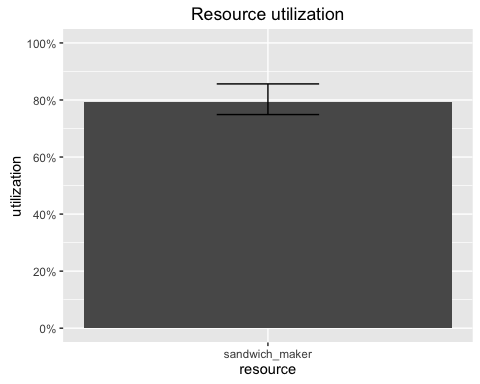
plot\_evolution\_arrival\_times(env, type="waiting\_time")



That had better be a great sandwich!! It might be ~70 minutes to get one! The black lines on the plot are the traces of the individual replications. The heavy blue line is the average.

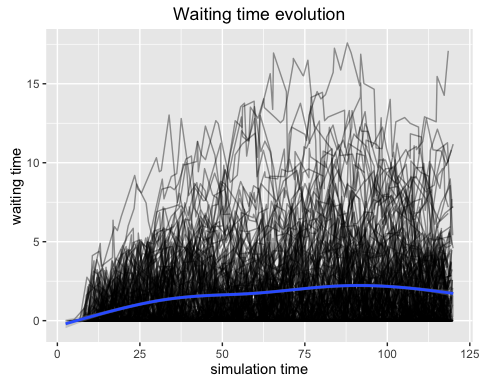
Let's see how we can reduce the wait. Since the sandwich\_maker has to wait a little bit of time for each sandwich to warm before finishing it they could be working on another sandwich during that time. Let's change the sandwich\_maker capacity to 3 (meaning they can be making up to 3 sandwiches at a time)

env <- lapply(1:200, function(i) {  
 simmer("Sandwich Joe") %>%  
 add\_resource("sandwich\_maker", 3) %>%  
 add\_generator("customer", t0, function() rpois(1, 2)) %>%  
 # limit run to 80 time units!  
 run(120)  
})  
plot\_resource\_utilization(env, c("sandwich\_maker"))



Now, the customer has to wait less time for their sandwich. But that does make the job of a sandwich\_maker pretty complex.

plot\_evolution\_arrival\_times(env, type="waiting\_time")



# Conclusion

This essay has demonstrated the power of simmer and simulation as a method for studying system dynamics. We investigated the dynamics of a sandwich shop with one sandwich\_maker. and found that the sandwich\_maker is a very busy person form 11 am to 1 pm.