Tratamiento Estadístico Computacional de la Información

UNIVERSIDAD
COMPLUTENSE
MADRID

POLITÉCNICA

Asignatura: Técnicas de Montecarlo

Práctica grupal

Realizado por:
Javier Castellano Soria
Ignacio Fernández Sánchez-Pascuala

6 de Diciembre de 2023, curso 2023/2024

Índice general

1. Problema Teoría de Colas

1.1. Enunciado del Problema: Cuestión a resolver e información relevante.

Cuestión a Resolver: Diseñar un modelo de simulación para una hamburguesería que incluya la producción de pan y carne, el montaje de hamburguesas, la atención a clientes y la venta de hamburguesas. El objetivo es analizar el rendimiento del sistema y optimizar el servicio para maximizar los beneficios.

Información Relevante:

1. **Configuración de la Hamburguesería:**
   - La hamburguesería cuenta con varios servidores (chefs) para el montaje de hamburguesas.
   - El tiempo de apertura de la hamburguesería es de 12 horas (720 minutos), de 9:00 a 21:00.
2. **Producción de Pan y Carne:**
   - La producción de pan y carne se realiza por separado (distintas fuentes) con un tiempo medio de elaboración de 7 minutos.
3. **Montaje de Hamburguesas:**
   - El servicio de montaje de las hamburguesas se realiza cada 8 minutos de media.
   - Se requiere tanto pan como carne para montar una hamburguesa.
   - Hay una cola para pan y otra para carne.
4. **Atención a Clientes:**
   - Los clientes llegan a la hamburguesería con una tasa de llegada de 1 cliente cada 7 minutos.
   - La tasa de servicio para realizar el pago de una hamburguesa en caja es de 1 cliente por minuto.
   - Hay una cola donde esperan los clientes que no pueden ser atendidos, ya sea porque la caja está ocupada o porque no haya hamburguesas hechas.
5. **Venta de Hamburguesas:**
   - Una vez montada, una hamburguesa puede ser vendida.
   - Existe una cola donde se almacenan las hamburguesas ya hechas, a la espera de ser compradas por un cliente.
6. **Tiempo Máximo en Cola:**

- Existe un tiempo máximo de 40 minutos en cola para las hamburguesas ya hechas antes de que se considere que la hamburguesa se ha enfriado y se tire.
- El tiempo máximo de espera de los clientes en la cola sigue una distribución *Gamma* de media 25 minutos.

7. **Otros Detalles:**
   - El precio de una hamburguesa es de 9.90 unidades monetarias.
   - Se tiene un costo asociado a la producción: sueldo de los chefs, costo del pan y costo de la carne.

Objetivos del Análisis:

1. Calcular medidas de desempeño del sistema, como el tiempo y el número medio de clientes, hamburguesas, panes y carnes en cola.
2. Analizar el tiempo y número medio de clientes y hamburguesas en el sistema.
3. Analizar el número medio de chefs ocupados en el sistema.
4. Evaluar el porcentaje de abandono de clientes y hamburguesas.
5. Calcular los ingresos, gastos y beneficios totales del servicio.
6. Probar diferentes configuraciones, como aumentar o disminuir el número de servidores, para ver su impacto en el desempeño del sistema.

Consideraciones Adicionales:

- La simulación se ejecuta hasta que se alcanza el tiempo de cierre de la hamburguesería.
- Al final del día, se atiende al último cliente en la caja, y se cierra la hamburguesería.

1.2. Modelo Conceptual de Simulación. Identificación de Elementos y Rutinas.

En la figura 1.1 se muestra el diagrama de simulación explicando los diferentes procesos que ocurren. Debajo de esta aparece una leyenda con el pseudocódigo correspondiente a cada celda de código del diagrama. Las simplificaciones del problemas que se han considerado son:

- No se ha considerado una capacidad máxima en el sistema para los panes, las carnes, las hamburguesas y los clientes.
- Independencia entre los tiempos de servicio y los tiempos de llegada.
- Se supone constante la tasa de clientes a lo largo de todo el día.
- Tiempos de servicio ilimitados.
- No se han contemplado posibles averías que impidan temporalmente el uso de algunos servicios.

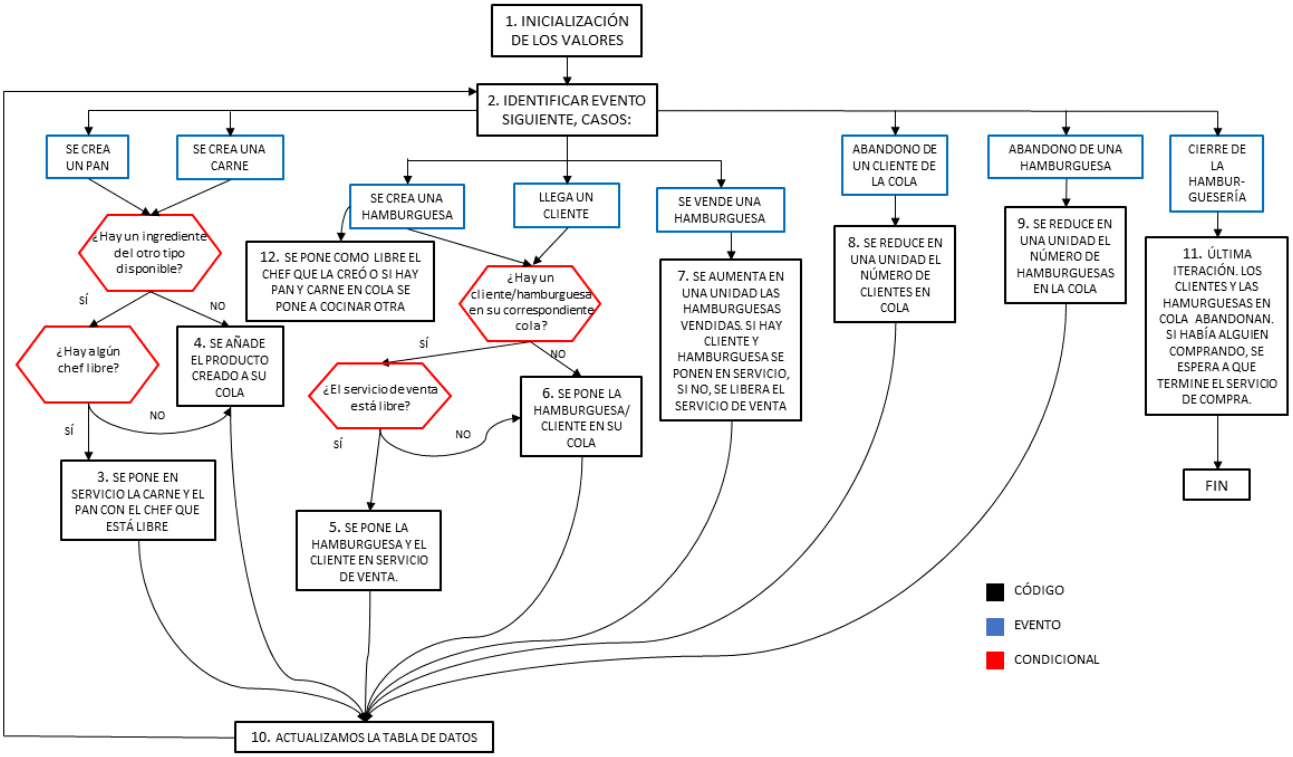

Figura 1.1: Diagrama de flujo

Elementos, rutinas y variables del modelo:

- Las variables de estado que describen la hamburguesería en cada momento son ColaPan (nº de panes en cola), ColaCarne (nº de carnes en cola), Servicio (nº de chefs que están dando servicio), ColaHamburguesa (nº de hamburguesas hechas en cola), ColaCliente (nº de clientes en cola para comprar), ServicioC (nº de pares cliente/hamburguesa en servicio de compra) y stay (tiempo que debe transcurrir para pasar al siguiente evento desde el actual).
- t es el tiempo de reloj.
- Los distintos eventos se identifican con un recuadro azul en el diagrama 1.1.
- Los contadores que aparecen en el código usados posteriormente para calcular ciertas cantidades son: Llegadas (es una lista de dos elementos que indica el nº de panes y carnes producidas), LlegadasC (nº de clientes que han llegado), LlegadasH (nº de hamburguesas producidas), AbandonoC (nº de clientes que abandonan por superar tiempo de espera), AbandonoH (nº de hamburguesas desechadas por enfriarse) y HamburguesasVendidas.
- La inicialización se corresponde al punto 1. del diagrama. La regla de parada consiste en comprobar si el siguiente evento, distinto del cierre, ocurre pasados 720 minutos desde la apertura. En este caso, el evento sería el cierre de la hamburguesería. El generador de resultados corresponde al *dataframe* en el que se va añadiendo la información de cada evento.
- Nservidores es el nº de chefs trabajando.

- nextLPan es el tiempo del próximo pan hecho y nextLCarne el de la carne.
- nextLCliente es el tiempo de la próxima llegada de un cliente al sistema.
- nextS es una lista de longitud Nservidores que indica el tiempo de finalización del montaje de la hamburguesa por cada chef. Si es Inf significa que el chef no está ocupado.
- nextSCliente es el tiempo de finalización del servicio de compra. Si es Inf indica que no hay nadie en servicio de compra.
- TmaxH es tiempo máximo que puede estar una hamburguesa en la cola.
- nextSColaH es una lista con los posibles tiempos de abandono de las hamburguesas que están en cola. Análogo para nextSColaC con los clientes.

Pseudocódigo:

1. t = 0; Tcierre = 720; Nservidores = 2; ColaPan = 0; ColaCarne = 0; Servicio = 0; Llegadas = [0, 0]; Generar LPan(Exp); nextLPan = t + LPan; Generar LCarne(Exp); nextLCarne = t + LCarne; nextS = rep(Inf, Nservidores); ColaHamburguesa = 0; LlegadasH = 0; ColaCliente = 0; ServicioC = 0; LlegadasC = 0; nextSCliente = Inf; Generar LCliente(Exp); nextLCliente = t + LCliente; TmaxH = 40; nextSColaH = [Inf]; nextSColaC = [Inf]; AbandonoH = 0; AbandonoC = 0; HamburguesasVendidas=0; stay = mínimo(nextLPan, nextLCarne, nextLCliente); Creo *dataframe* con los datos actuales;
2. evento, t = argumento-mínimo(nextLPan, nextLCarne, mínimo(nextS), nextLCliente, nextSCliente, mínimo(nextSColaH), mínimo(nextSColaC)); **Si** t>Tcierre **hacer:** evento = cierre; t = Tcierre; **fin**; .
3. **Si** llega un pan **hacer:** Llegadas[1] = Llegadas[1]+1; ColaCarne = ColaCarne - 1; i = min(which(isInf(nextS))); Generar auxnextS(Exp); nexts[i] = t + auxnextS); Servicio = Servicio + 1; Generar LPan(Exp); nextLPan = t + LPan; **fin**; **Si** llega una carne **hacer:** Llegadas[2] = Llegadas[2]+1; ColaPan = ColaPan - 1; i = min(which(is.nan(nextS))); Generar auxnextS(Exp); nexts[i] = t + auxnextS; Servicio = Servicio + 1; Generar LCarne(Exp); nextLCarne= t + LCarne; **fin** .
4. **Si** llega un pan **hacer:** Llegadas[1] = Llegadas[1]+1; ColaPan = ColaPan + 1; Generar LPan(Exp); nextLPan = t + LPan; **fin**; **Si** llega una carne **hacer:** Llegadas[2] = Llegadas[2]+1; Colacarne = ColaCarne + 1; Generar LCarne(Exp); nextLCarne= t + LCarne; **fin**; .
5. **Si** se crea una hamburguesa **hacer:** ColaC = ColaC - 1; ServicioC = ServicioC + 1; Elimino el tiempo de abandono por espera del cliente en nextSColaC; Generar auxnextSCliente(Exp); nextSCliente = t + auxnextSCliente; **fin**; **Si** llega un cliente **hacer:** LlegadasC = LlegadasC+1; Generar auxnextLCliente(Exp); nextLCliente = t + auxnextLCliente; ColaHamburguesa = ColaHamburguesa - 1; Elimino el tiempo de abandono por enfriamiento de hamburguesa en nextSColaH; ServicioC = ServicioC + 1; Generar auxnextSCliente(Exp); nextSCliente = t + auxnextSCliente; **fin**.
6. **Si** se crea una hamburguesa **hacer:** ColaHamburguesa = ColaHamburguesa + 1; añadir (nextSColaH, t + TmaxH); **fin**; **Si** llega un cliente **hacer:** LlegadasC = LlegadasC+1;

Generar auxnextLCliente(Exp); nextLCliente = t + auxnextLCliente; ColaCliente = ColaCliente + 1; Generar espera(Gamma); añadir(nextSColaC, t + espera); **fin**; .

7. HamburguesasVendidas = HamburguesasVendidas + 1; **Si** ColaClientes>0 y ColaHamburguesas>0 **hacer:** ColaHamburguesa = ColaHamburguesa - 1; ColaCliente = ColaCliente - 1; Elimino el tiempo de abandono por enfriamiento de hamburguesa en nextSColaH; Elimino el tiempo de abandono por espera del cliente en nextSColaC; Generar auxnextSCliente(Exp); nextSCliente = t + auxnextSCliente; **Si no hacer:** ServicioC = ServicioC - 1; nextSCliente = Inf; **fin**; .
8. ColaCliente = ColaCliente - 1; Elimino el tiempo de abandono por espera del cliente en nextSColaC;
9. ColaHamburguesa = ColaHamburguesa - 1; Elimino el tiempo de abandono por enfriamiento de hamburguesa en nextSColaH;
10. Se halla stay que es el lapso de tiempo hasta el siguiente evento (incluido el tiempo de cierre) y se añade al *dataframe* los datos correspondientes al evento.; .
11. stay = 0; **Si** ServicioC>0 **hacer:** HamburguesasVendidas = HamburguesasVendidas + 1; stay = nextSCliente - Tcierre **fin**; AbandonoC = AbandonoC + ColaCliente; ColaCliente=0; nextSColaC=[Inf]; AbandonoH = AbandonoH + ColaH; ColaHamburguesa=0; nextSColaH=[Inf]; Añadimos la iformación de la última iteración al *dataframe*; .
12. LlegadasH = LlegadasH+1; i = which(t == nextS); **Si** ColaPan>0 y ColaCarne>0 **hacer:** ColaPan = ColaPan - 1; ColaCarne = ColaCarne - 1; Generar auxnextS(Exp); nextS[i] = t + auxnextS; **Si no hacer:** Servicio = Servicio - 1; nextS[i] = Inf; **fin**;

1.3. Modelo Computacional.

El modelo computacional para la simulación de la hamburguesería se implementó en el lenguaje de programación R. En el Anexo se añade el código utilizado para la simulación, junto con la correcta documentación del mismo mediante comentarios e indicaciones en el código.

1.4. Análisis resultados de la simulación.

1.4.1. Resultados con 2 Chefs

A continuación, se presentan los resultados obtenidos al simular la hamburguesería durante 1 mes (20 días laborables) con 2 Chefs:

Métrica	**Valor Medio del día**
Tiempo medio clientes en cola (Wq_C)	8.63 minutos
Tiempo medio hamburguesas en cola (Wq_H)	11.74 minutos
Tiempo medio panes en cola (Wq_P)	22.94 minutos
Tiempo medio carnes en cola (Wq_{Carn})	29.07 minutos
Número medio clientes en cola (Lq_C)	1.25 clientes
Número medio hamburguesas en cola (Lq_H)	1.63 hamburguesas
Número medio panes en cola (Lq_P)	3.38 panes
Número medio carnes en cola (Lq_{Carn})	4.46 carnes
Tiempo medio clientes en el sistema (W_C)	9.51 minutos
Numero medio clientes en el sistema (L_C)	1.38 clientes
Tiempo medio hamburguesas en el sistema (W_H)	12.67 minutos
Numero medio de hamburguesas en el sistema (L_H)	1.76 hamburguesas
Numero medio de chefs ocupados en el sistema	1.09 chefs
Porcentaje abandono de clientes	13.6 %
Porcentaje abandono de hamburguesas	9.3 %
Ingresos totales	$874.17
Gastos totales	$516.15
Beneficios totales	$358.02

Cuadro 1.1: Resultados de la simulación con 2 chefs.

Como se observa en la tabla, el tiempo medio de panes y carnes en cola es muy elevado. La incorporación de otro chef podría ayudar a reducir este tiempo, y así poder crear hamburguesas más rápido. Esto también reducirá el porcentaje de clientes que abandonan la cola al ser desatendidos y reducirá su tiempo de espera.

Sin embargo, también puede provocar que se creen más hamburguesas de la demanda en cada momento, aumentando el porcentaje de hamburguesas desechadas y el tiempo medio de hamburguesas en cola, que también ya es bastante elevado. Además, el número medio de chefs ocupados no es tan elevado y la incorporación de chefs supone un mayor gasto.

Se puede observar que el modelo está produciendo beneficios, y que además el porcentaje de abandono de clientes y hamburguesas no es tan grande, aunque sí mejorable.

Veamos en el siguiente apartado qué beneficios obtendría la hamburguesería tras la contratación o destitución de chefs.

1.4.2. Variación de Beneficios

Se presenta la variación de beneficios al cambiar el número de servidores/chefs de nuestra hamburguesería:

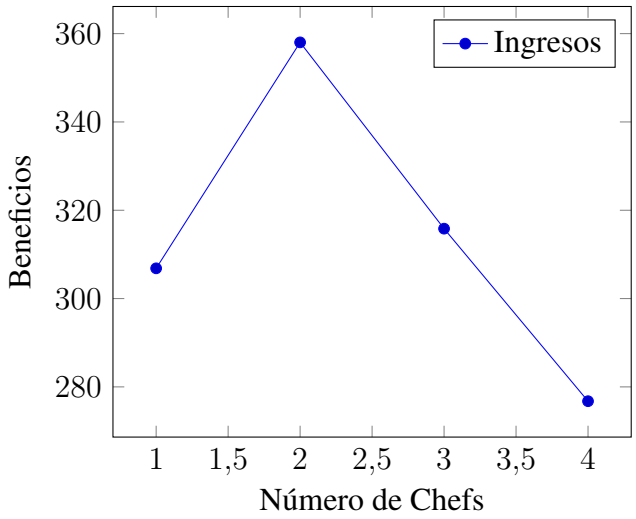

Figura 1.2: Variación de beneficios según el número de servidores.

Como vemos en la gráfica, el número óptimo de chefs en nuestra hamburguesería son 2, ya que es el que mayores beneficios consigue de media al día y el que mejor equilibrio consigue entre oferta y demanda.

Para 1 chef, los beneficios son menores, seguramente debido a la incapacidad de atender a tiempo a la demanda, que provoca que muchos clientes abandonen.

Por otro lado, con 3 y 4 chefs los beneficios también decrecen, esto puede ser debido a la producción de más hamburguesas de las requeridas en cada momento, provocando que muchas hamburguesas tengan que ser desechadas, y un mayor gasto en la contratación de chefs.

2. Anexo: Código R.

```r
#Cargar bibliotecas
library(dplyr);
library(tidyr);

#Semilla aleatoria
set.seed(1234)

#PARAMETROS SIMULACION:
n_simulaciones <- 20; #Numero dias laborables en un mes
n_servidores <- 2; #Numero de servidores/chefs
Hora = 720; # Tiempo de apertura hamburgueseria: 9:00 - 21:00

lambdaL1 = 1/7.; #Tiempo Prod pan cada 7 minutos
lambdaL2 = 1/7.; #Tiempo Prod carne
lambdaL<-c(lambdaL1,lambdaL2);
lambdaS <- 1/8.; # Tasa servicio chefs: cada 8 minutos (
   montaje hamburguesa)

lambdaLC <- 1/7. # Tasa Llegada cliente
lambdaSC <- 1/1. # Tasa servicio de pago para adquirir la
   hamburguesa

#Tiempos maximo en cola para las hamburguesas y clientes->
TmaxH = 40; #Tiempo maximo hamburguesas en cola antes de
   desecharla

#El tiempo abandono clientes cola va a seguir una Gamma de
   media 25:
MediaC <- 25;
formaC<- 5;
escalaC<- MediaC / formaC;

#LISTAS PARA ALMACENAR LOS RESULTADOS POR SIMULACION:
beneficios_por_dia = c()
gastos_por_dia = c()
ingresos_por_dia = c()
Tiempo_medio_clientes_cola_por_dia = c()
Tiempo_medio_hamburguesas_cola_por_dia = c()
Tiempo_medio_panes_cola_por_dia = c()
Tiempo_medio_carnes_cola_por_dia = c()
Numero_medio_clientes_cola_por_dia = c()
Numero_medio_hamburguesas_cola_por_dia = c()
Numero_medio_panes_cola_por_dia = c()
Numero_medio_carnes_cola_por_dia = c()
```

```
Tiempo_medio_clientes_sistema_por_dia = c()
Numero_medio_clientes_sistema_por_dia = c()
Numero_medio_chefs_ocupados_por_dia = c()
Tiempo_medio_hamburguesas_sistema_por_dia = c()
Numero_medio_hamburguesas_sistema_por_dia = c()
Porcentaje_abandono_clientes_por_dia = c()
Porcentaje_abandono_hamburguesas_por_dia = c()

#FUNCIONES AUXILIARES SIMULACION DETECCION SIGUIENTE EVENTO
es_llegada_pan_carne <-function(nextS, nextL, n_servidores,
   nextSC, nextLC,nextSColaC,nextSColaH){
  return( ((sum(is.nan(nextS)) == n_servidores) ||
             (min(nextL) <= min(nextS, na.rm = TRUE))) &&
            (min(nextL) <= min(nextLC,nextSC,nextSColaC,
                               nextSColaH, na.rm = TRUE)) )
}

es_montaje <-function(nextS, nextL, n_servidores, nextSC,
   nextLC, nextSColaC, nextSColaH){
  return( (sum(is.nan(nextS)) != n_servidores) &&
            min(nextS, na.rm = TRUE) < min(nextL, nextLC,
               nextSColaH,nextSColaC, nextSC, na.rm = TRUE) )
}

es_llegada_cliente <- function(nextS, nextL, nextSC, nextLC,
   nextSColaC, nextSColaH){
  return( nextLC <= min(nextL, nextS, nextSC, nextSColaH,
     nextSColaC, na.rm = TRUE) )
}

es_venta_hamburguesa <- function(nextS, nextL, nextSC, nextLC
   , nextSColaC,nextSColaH){
  return((!is.nan(nextSC)) && (nextSC <= min(nextL, nextS,
     nextLC,nextSColaH,nextSColaC, na.rm = TRUE)))
}

es_abandono_cliente <- function(nextS, nextL, nextSC, nextLC,
    nextSColaC, nextSColaH){
  return( min(nextSColaC,na.rm=T) <= min(nextL, nextS, nextLC
     ,nextSColaH, nextSC, na.rm = TRUE) )
}

#ITERACION SOBRE EL NUMERO DE DIAS SIMULADOS:
for (i in 1:n_simulaciones){

  #INICIALIZACION VARIABLES:
  t = 0;
```

```
# Colas y servicios para crear las hamburguesas->
Cola1 = 0; #Numero panes en cola
Cola2 = 0; #Numero carnes en cola
Servicio = 0;  #Numero de hamburguesas haciendose por el chef
Llegadas = c(0, 0); #Numero de panes y carnes producidas.
Cola<-c(Cola1,Cola2);
nextL1 = rexp(1, rate = lambdaL1);  #Proximo pan hecho
nextL2=rexp(1,rate=lambdaL2);     #Proxima carne hecha
nextL <- c(nextL1, nextL2);
nextS = rep(NaN, n_servidores); #Proxima salida hamburguesa hecha
Hamburguesas_vendidas <- 0;

#Cola y servicio para la compra de Hamburguesas->
ColaH = 0; # Hamburguesas en cola
ColaC = 0; # Clientes en cola
ServicioC = 0;  #Clientes en servicio
LlegadasC = 0;  #Numero llegadas Clientes
LlegadasH = 0;  #Numero de Hamburguesas hechas
nextLC = rexp(1, rate = lambdaLC);  #Proxima llegada cliente
nextSC = NaN;  #proxima salida cliente
nextSColaH <- NaN  #Proximos tiempos de abandono cola hamburguesas
nextSColaC <- NaN  #Proximos tiempos de abandono cola clientes
AbandonoC <- 0;  #Numero de clientes que han abandonado la cola
AbandonoH <- 0;  #Numero de hamburguesas desechadas

stay = min(nextL, nextLC); #Estado inicial

#Creamos tibble para almacenar datos:
data_Hamburgueseria <- tibble(
  t, ColaH, ColaC, ServicioC, LlegadasC, LlegadasH, nextLC, nextSC, Cola=list(Cola), Servicio, Llegadas = list(Llegadas),
  stay, nextL=list(nextL), nextS = list(nextS),
  nextSColaC =list(nextSColaC), nextSColaH=list(nextSColaH),
  AbandonoC, AbandonoH, Hamburguesas_vendidas) %>%
  unnest_wider(nextS, names_sep = '_')  %>%
  unnest_wider(nextL, names_sep = '_') %>%
  unnest_wider(Cola, names_sep = '_') %>%
  unnest_wider(Llegadas, names_sep = '_')

```

```
#### SIMULACION EVENTO####

while (min(c(nextL, nextS, nextLC, nextSC, nextSColaC,
   nextSColaH), na.rm = TRUE) <= Hora) {
  #Evento: Llegada de pan y carne
  if (es_llegada_pan_carne(nextS, nextL, n_servidores,
     nextSC, nextLC, nextSColaC,nextSColaH)) {
    # Identificar la llegada mas rapida
    i <- which.min(nextL)
    t <- nextL[i]
    Llegadas[i] <- Llegadas[i] + 1

    #Calcular nuevo instante de llegadas
    nextL[i] <- t + rexp(1, rate = lambdaL[i])

    # Determinar el indice del otro producto (1 si i=2, 2
       si i=1)
    j <- 3-i

    if (Servicio >= n_servidores || Cola[j] == 0) {
      # Si no esta el otro producto disponible o el
         servicio completo
      Cola[i] <- Cola[i] + 1

    } else {
      # Si ambos productos estan disponibles, pasar a
         servicio
      Servicio <- Servicio + 1
      Cola[j] <- Cola[j] - 1

      if (sum(is.nan(nextS)) == n_servidores) {
        # Si todos los servidores estan vacios, asignar al
           primer servidor
        nextS[1] <- t + rexp(1, rate = lambdaS)

      } else {
        # Si algun servidor esta libre, asignar al servidor
           libre
        i_serv <- min(which(is.nan(nextS)))
        nextS[i_serv] <- t + rexp(1, rate = lambdaS)
      }
    }

  }
  #Evento: Montaje de Hamburguesa
  else if (es_montaje(nextS, nextL, n_servidores, nextSC,
     nextLC, nextSColaC, nextSColaH)){
```

```
156       # Identificar el instante en el que se produce la
              salida
157       t = min(nextS, na.rm = T);
158       i_serv <- min(which(t == nextS)) #Servidor en el que se
              produce la salida
159
160       if (Cola[1] > 0 && Cola[2] > 0){ # Si hay pan y carne
              en cola
161         # Utilizar el pan y la carne, quitarlos de la cola
                actualizar
162         #el tiempo salida de servicio
163         Cola[1] = Cola[1] - 1;
164         Cola[2]=Cola[2] - 1;
165         nextS[i_serv] <- t + rexp(1, rate = lambdaS);
166
167       } else { # Si no hay pan y carne
168         # Decrementar las hamburguesas en servicio y
169         #marcar el servidor como disponible
170         Servicio = Servicio - 1;
171         nextS[i_serv] <- NaN;
172       }
173
174       LlegadasH = LlegadasH+1;
175
176       # Determinar si colocar la hamburguesa en cola o
              venderla directamente
177       if (ColaC == 0 || ServicioC >= 1){
178         # Colocar la hamburguesa en cola
179         ColaH = ColaH+1;
180         TFria <- t + TmaxH;
181         nextSColaH <- c(nextSColaH,TFria);
182
183       } else{
184         # Vender la hamburguesa directamente
185         ColaC = ColaC-1;
186         TminAbandono <- min(nextSColaC, na.rm=T);
187         nextSColaC <- nextSColaC[nextSColaC != TminAbandono];
188         ServicioC = ServicioC+1;
189         nextSC <- t + rexp(1, rate = lambdaSC);
190       }
191
192     }
193     #Evento: Llegada cliente
194     else if (es_llegada_cliente(nextS, nextL, nextSC, nextLC,
            nextSColaC, nextSColaH)){
195       # Actualizar el tiempo actual y calcular el proximo
196       #instante de llegada de cliente
```

```
197       t = nextLC;
198       nextLC = t + rexp(1, rate = lambdaLC);
199
200       LlegadasC = LlegadasC+1;
201
202       if (ColaH == 0 || ServicioC >= 1){
203         # Si no hay hamburguesas disponibles o hay al menos
              una en servicio,
204         #el cliente se une a la cola de clientes
205         ColaC = ColaC+1;
206         TAbandonoC <- t+  rgamma(n =1 , shape = formaC, scale
              = escalaC);
207         nextSColaC <- c(nextSColaC, TAbandonoC)
208         rm(TAbandonoC)
209
210       } else {
211         # Si hay hamburguesas disponibles, se procede a la
              venta directa
212         ColaH = ColaH-1;
213         TminFria <- min(nextSColaH, na.rm=T);
214         nextSColaH <- nextSColaH[nextSColaH != TminFria];
215         ServicioC = ServicioC+1;
216         nextSC <- t + rexp(1, rate = lambdaSC);
217       }
218     }
219     #Evento: Servicio venta de hamburguesas
220     else if(es_venta_hamburguesa(nextS, nextL, nextSC, nextLC
          , nextSColaC,nextSColaH)){
221       # Actualizar el tiempo actual y registrar la venta de
            una hamburguesa
222       t = nextSC;
223       Hamburguesas_vendidas<-Hamburguesas_vendidas+1;
224
225       if (ColaC > 0 && ColaH > 0){
226         # Si hay clientes y hamburguesas en cola, entra otro
              a servicio de caja
227         ColaC = ColaC-1;
228         TminAbandono <- min(nextSColaC, na.rm=T);
229         nextSColaC <- nextSColaC[nextSColaC != TminAbandono];
230         ColaH = ColaH-1;
231         TminFria <- min(nextSColaH, na.rm=T);
232         nextSColaH <- nextSColaH[nextSColaH != TminFria];
233         # Establecer el proximo instante de servicio
234         nextSC = t + rexp(1, rate = lambdaSC);
235
236       } else {
```

```
237         #Si no hay clientes o hamburguesas hechas, no se
               entra a servicio de caja
238         ServicioC = 0;
239         nextSC = NaN;
240       }
241     }
242     #Evento: Abandono de cliente
243     else if(es_abandono_cliente(nextS, nextL, nextSC, nextLC,
            nextSColaC, nextSColaH)){
244       # Reducir la cola de clientes y registrar el abandono
             de un cliente
245       ColaC=ColaC-1;
246       AbandonoC <- AbandonoC+1;
247
248       # Identificar el proximo instante de abandono y
249       #actualizar el tiempo actual
250       TminAbandono <- min(nextSColaC, na.rm=T);
251       t <- TminAbandono;
252       nextSColaC <- nextSColaC[nextSColaC != TminAbandono];
253     }
254     #Evento: Hamburguesa fria y abandono de la cola
255     else{
256       # Reducir la cola de hamburguesas y registrar
257       # el abandono de una hamburguesa
258       ColaH=ColaH-1;
259       AbandonoH <- AbandonoH +1;
260
261       # Identificar el proximo instante de abandono y
262       #actualizar el tiempo actual
263       TminFria <- min(nextSColaH, na.rm=T);
264       t <- TminFria;
265       nextSColaH <- nextSColaH[nextSColaH != TminFria];
266     }
267
268     stay = min(nextL,nextS,nextSC,nextLC,nextSColaC,
              nextSColaH, Hora, na.rm=T)-t;
269
270     #Actualizamos el tibble tras la simulacion del evento.
271     data_Hamburgueseria <- rbind(
272       data_Hamburgueseria,
273       tibble(
274         t, ColaH, ColaC, ServicioC, LlegadasC, LlegadasH,
              nextLC, nextSC,
275         Cola=list(Cola), Servicio, Llegadas=list(Llegadas),
              stay,
276         nextL=list(nextL), nextS = list(nextS),nextSColaC =
              list(nextSColaC),
```

```
277         nextSColaH = list(nextSColaH), AbandonoC, AbandonoH,
278         Hamburguesas_vendidas)
279       %>%
280         unnest_wider(nextS, names_sep = '_') %>%
281         unnest_wider(nextL, names_sep = '_') %>%
282         unnest_wider(Cola, names_sep = '_') %>%
283         unnest_wider(Llegadas, names_sep = '_')
284     )
285   }
286
287   #ULTIMA ITERACION CIERRE:
288
289   #Si hay justo un cliente comprando al cerrar, le damos la
        hamburguesa
290   #y cerramos. Los clientes y hamburguesas que se quedan en
        cola los
291   #consideramos como que abandonan.
292
293   stay = 0;
294   if (ServicioC>0){
295     Hamburguesas_vendidas<-Hamburguesas_vendidas+1;
296     stay = nextSC - Hora;
297   }
298
299   if (ColaC>0){
300     AbandonoC<-AbandonoC+ColaC;
301     ColaC<-0;
302     nextSColaC<-NaN;
303   }
304
305   if (ColaH>0){
306     AbandonoH<-AbandonoH+ColaH;
307     ColaH<-0;
308     nextSColaH<- NaN;
309   }
310   t = Hora;
311
312   #Actualizacion final tibble:
313   data_Hamburgueseria <- rbind(
314     data_Hamburgueseria,
315     tibble(
316       t, ColaH, ColaC, ServicioC, LlegadasC, LlegadasH,
          nextLC, nextSC,
317       Cola=list(Cola), Servicio, Llegadas=list(Llegadas),
          stay,
318       nextL=list(nextL), nextS = list(nextS),nextSColaC =
          list(nextSColaC),
```

```
      nextSColaH = list(nextSColaH), AbandonoC, AbandonoH,
      Hamburguesas_vendidas)
  %>%
    unnest_wider(nextS, names_sep = '_') %>%
    unnest_wider(nextL, names_sep = '_') %>%
    unnest_wider(Cola, names_sep = '_') %>%
    unnest_wider(Llegadas, names_sep = '_')
)

#CALCULO Y ACUMULACION ESTADISTICAS SIMULACION UN DIA
n_eventos <- length(data_Hamburgueseria$t)
n_clientes <- data_Hamburgueseria$LlegadasC[n_eventos]
n_hamburguesas <- data_Hamburgueseria$LlegadasH[n_eventos]
n_panes <- data_Hamburgueseria$Llegadas_1[n_eventos]
n_carnes <- data_Hamburgueseria$Llegadas_2[n_eventos]
tiempo_total <- data_Hamburgueseria$t[n_eventos]

# 1) Tiempo medio clientes en cola
WqC <- sum(data_Hamburgueseria$ColaC * (data_Hamburgueseria
   $stay/n_clientes))
Tiempo_medio_clientes_cola_por_dia<-c(Tiempo_medio_clientes
   _cola_por_dia,WqC)

# 2) Tiempo medio hamburguesas en cola
WqH <- sum(data_Hamburgueseria$ColaH * (data_Hamburgueseria
   $stay/n_hamburguesas))
Tiempo_medio_hamburguesas_cola_por_dia<-
  c(Tiempo_medio_hamburguesas_cola_por_dia,WqH)

# 3) Tiempo medio panes en cola
WqP <- sum(data_Hamburgueseria$Cola_1 * (data_
   Hamburgueseria$stay/n_panes))
Tiempo_medio_panes_cola_por_dia<-c(Tiempo_medio_panes_cola_
   por_dia,WqP)

# 4) Tiempo medio carnes en cola
WqCarn <- sum(data_Hamburgueseria$Cola_2 * (data_
   Hamburgueseria$stay/n_carnes))
Tiempo_medio_carnes_cola_por_dia<-c(Tiempo_medio_carnes_
   cola_por_dia,WqCarn)

# 5) Porcentaje abandono de clientes
n_abandonos_clientes <- data_Hamburgueseria$AbandonoC[n_
   eventos]
porcentaje_abandono_clientes <- n_abandonos_clientes/n_
   clientes
```

```
356 Porcentaje_abandono_clientes_por_dia<-c(Porcentaje_abandono
       _clientes_por_dia, porcentaje_abandono_clientes)
357
358 # 6) Porcentaje hamburguesas perdidas
359 n_abandonos_hamburguesas <- data_Hamburgueseria$AbandonoH[n
       _eventos]
360 porcentaje_abandono_hamburguesas <- n_abandonos_
       hamburguesas/n_hamburguesas
361 Porcentaje_abandono_hamburguesas_por_dia<-
362   c(Porcentaje_abandono_hamburguesas_por_dia,
363     porcentaje_abandono_hamburguesas)
364
365 # 7) Ingresos totales
366 precio_ham <- 9.90
367 n_hamburguesas_vendidas <- data_Hamburgueseria$Hamburguesas
       _vendidas[n_eventos]
368 ingresos <- n_hamburguesas_vendidas*precio_ham
369 ingresos_por_dia <- c(ingresos_por_dia,ingresos)
370
371 # 8) Gastos totales
372 sueldo_chef <- 50
373 coste_pan <- 1
374 coste_carne <- 3
375 gastos <- sueldo_chef*n_servidores + coste_pan*n_panes +
       coste_carne*n_carnes
376 gastos_por_dia <- c(gastos_por_dia, gastos)
377
378 # 9) Beneficios totales
379 beneficios <- ingresos - gastos
380 beneficios_por_dia <- c(beneficios_por_dia, beneficios)
381
382 # 10) Media clientes en cola
383 LqC <- sum(data_Hamburgueseria$ColaC*          (data_
       Hamburgueseria$stay/tiempo_total))
384 Numero_medio_clientes_cola_por_dia<-c(Numero_medio_clientes
       _cola_por_dia,LqC)
385
386 # 11) Media hamburguesas en cola
387 LqH <- sum(data_Hamburgueseria$ColaH*(data_Hamburgueseria$
       stay/tiempo_total))
388 Numero_medio_hamburguesas_cola_por_dia<-
389   c(Numero_medio_hamburguesas_cola_por_dia,LqH)
390
391 # 12) Media panes en cola
392 LqP <- sum(data_Hamburgueseria$Cola_1*         (data_
       Hamburgueseria$stay/tiempo_total))
```

```
393   Numero_medio_panes_cola_por_dia<-c(Numero_medio_panes_cola_
          por_dia,LqP)
394
395   # 13) Media carnes en cola
396   LqCarn <- sum(data_Hamburgueseria$Cola_2*              (data
          _Hamburgueseria$stay/tiempo_total))
397   Numero_medio_carnes_cola_por_dia<-c(Numero_medio_carnes_
          cola_por_dia,LqCarn)
398
399   # 14) Tiempo medio clientes en el sistema
400   WC <- sum((data_Hamburgueseria$ColaC + data_Hamburgueseria$
          ServicioC)*data_Hamburgueseria$    stay/n_clientes)
401   Tiempo_medio_clientes_sistema_por_dia<-
402     c(Tiempo_medio_clientes_sistema_por_dia,WC)
403
404   # 15) Numero medio clientes en el sistema
405   LC <- sum((data_Hamburgueseria$ColaC + data_Hamburgueseria$
          ServicioC)                          *data_Hamburgueseria$
          stay/tiempo_total)
406   Numero_medio_clientes_sistema_por_dia<-
407     c(Numero_medio_clientes_sistema_por_dia,LC)
408
409   # 16) Tiempo medio de hamburguesas en el sistema
410   WH <- sum((data_Hamburgueseria$ColaH + data_Hamburgueseria$
          ServicioC)
411     *data_Hamburgueseria$stay/n_hamburguesas)
412   Tiempo_medio_hamburguesas_sistema_por_dia<-
413     c(Tiempo_medio_hamburguesas_sistema_por_dia,WH)
414
415   # 17) Numero medio de hamburguesas en el sistema
416   LH <- sum((data_Hamburgueseria$ColaH + data_Hamburgueseria$
          ServicioC)*
417         data_Hamburgueseria$stay/tiempo_total)
418   Numero_medio_hamburguesas_sistema_por_dia<-
419     c(Numero_medio_hamburguesas_sistema_por_dia,LH)
420
421   #18) Numero medio de servidores ocupados en el sistema
422   Media_chefs_ocupados <- sum(data_Hamburgueseria$Servicio*
423     (data_Hamburgueseria$stay/tiempo_total))
424   Numero_medio_chefs_ocupados_por_dia<-c(Numero_medio_chefs_
          ocupados_por_dia,Media_chefs_ocupados)
425 }
```

Extracto de código 2.1: Codigo R