M|M|1

Corsetti, Ornela Milagros[a], Golzman, Gabriel[a], Bengoechea, Guadalupe María[a]

[a] *Filiación de los autores*

Enunciado

En este trabajo práctico realizaremos la simulación de un modelo con un servidor y una cola (M|M|1), para posteriormente compararlo con una calculadora determinística, que se encuentra en el siguiente link `https://www.supositorio.com/rcalc/rcalclite_esp.htm`, de dicha simulación para comprobar su validez.

Palabras Clave:
Simulación, exponencial, arribos, partidas, FIFO, M|M|1, modelo, aleatorio.

Índice

Correos Electrnicos: `ocorsetti@frro.utn.edu.ar` (Corsetti, Ornela Milagros), `ggolzman@frro.utn.edu.ar` (Golzman, Gabriel), `guadabengoechea@gmail.com` (Bengoechea, Guadalupe María)

1. Marco Teórico.

1.1. Introducción.

En la *Teoría de Colas*, una disciplina dentro de la *Teoría matemática de la Probabilidad*, una cola M|M|1, representa la longitud de la cola en un sistema que posee un solo servidor, donde los arribos están determinados por un proceso de *Poisson* y los tiempos de servicio tienen una distribución exponencial.

Este modelo es el más elemental de los modelos de cola, y es de interés como caso de estudio de donde se pueden obtener muchas métricas significativas.

Una extensión de este modelo con más de un solo servidor es el modelo M|M|c.

1.2. Definición del modelo.

Una cola M|M|1 es un proceso estocástico cuyo espacio estacionario es el set {0,1,2,3,...} donde el valor corresponde al número de clientes en el sistema, incluyendo los que se encuentran en servicio en ese momento.

- Los arribos ocurren con un ratio de λ de acuerdo al proceso de *Poisson* moviéndose de i a $i + 1$.
- Los tiempos de servicio corresponden a una distribución exponencial con parámetro μ, donde $\frac{1}{\mu}$ es la media del tiempo de servicio.
- Un único servidor da servicio a los clientes, de a uno a la vez, de acuerdo al algoritmo FIFO, *first in, first out*. La disponibilidad del mismo se verifica a través del estado en el que el mismo se encuentra, sea este *disponible* u *ocupado*.
  Cuando el servicio se completa el cliente deja la cola y el número de clientes se reduce en uno.
- El buffer es de tamaño infinito, de manera que no hay un limite en la cantidad de clientes que puede contener.

1.3. Componentes.

- Población de clientes.
  - Conjunto de posibles clientes.
- Proceso de llegadas.
  - La forma en que llegan los clientes.
- Proceso de colas.
  - La disciplina de la cola: FIFO,LIFO,Prioridades
- Proceso de servicio.
  - La velocidad con la que el servidor atiende a los clientes.
  - Otros recursos que sean necesarios para que el servidor pueda prestar el servicio.
  - Definición de interrupciones.
- Procesos de salidas.
  - Clientes que parten definitivamente del sistema.
  - Clientes que pasan a otros servidores.

1.4. Medidas de rendimiento.

- Demora promedio por cliente.
- Proporción en que el servidor permanece ocupado.
- Número promedio en el tiempo de clientes en cola.

Si bien estas son las medidas de rendimiento más habituales, al estudiar un sistema se puede estudiar también el valor mínimo, el valor máximo o la distribución de probabilidad (comportamiento) de la variable.

Las medidas de rendimiento se determinan con el fin de **evaluar las distintas alternativas** de una experiencia de simulación o para **analizar el comportamiento** de un sistema bajo un grupo de condiciones dadas.

Son variables que se calculan a partir de los valores de los contadores estadísticos. Estos son las variables auxiliares dentro del algoritmo de simulación que permite calcular las medidas de rendimiento.

1.4.1. Demora promedio por cliente.

Es la demora en cola que en promedio esperan los clientes. Es una medida que indica desde el punto de vista del cliente qué tan bien se comporta el sistema al prestarles servicio.

Se calcula como la sumatoria de todas las demoras individuales por cliente, dividido la cantidad de clientes que completaron demora. En general se calcula para cada cola que haya en el sistema.

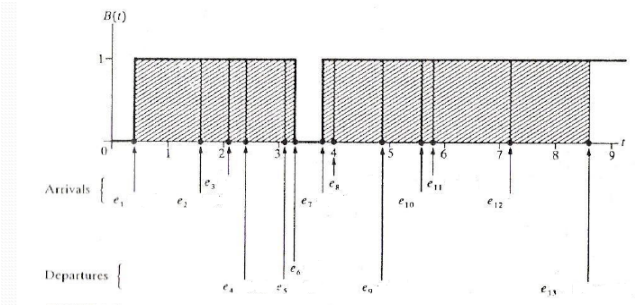
$$d(n) = \frac{\sum_{(i=1)}^{n} D_i}{n}$$

1.4.2. Proporción del tiempo en que el servidor permanece ocupado.

- Se calcula como la sumatoria de los tiempos de servicio incurridos en atención de clientes, dividido por el tiempo final de la simulación corrida.
- Se calcula para cada servidor en el sistema.
- Se puede ver como el área bajo $B(t)$ dividida por el tiempo de simulación.
- $B(t)$ es una función que asume el valor 1 desde el instante en que el servidor esta ocupado y 0 cuando esta desocupado.
- Esta variable evalúa el sistema desde el punto de vista de maximizar la utilización de los recursos sobre los que se ha invertido.

$$u(n) = \frac{\text{Tiempo de servicio acumulado}}{\text{Tiempo total}}$$

Figura 1: Gráfica de la B(t)



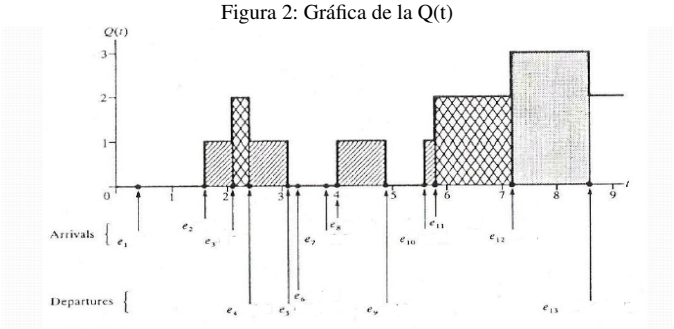
1.4.3. Numero promedio en el tiempo de clientes en cola.

- Representa el tamaño promedio de la cola a lo largo de toda la simulación.
- Se calcula a partir del área bajo $Q(t)$ dividido por el tiempo final de la simulación o de la corrida.
- $Q(t)$ es una función que indica la cantidad de clientes en cola en el tiempo t.
- $Q(t)$ se implementa en el algoritmo como una variable que se incrementa en 1 cuando entra un cliente en cola y se disminuye en 1 cuando se quita un cliente en cola.
- Esta variable evalúa el sistema desde el punto de vista de la atención al cliente.

$$q(n) = \frac{\text{Demora acumulada}}{\text{Cantidad de clientes que completaron demora}}$$

$$q(n) = \sum_{i=0}^{\infty} iP_i$$

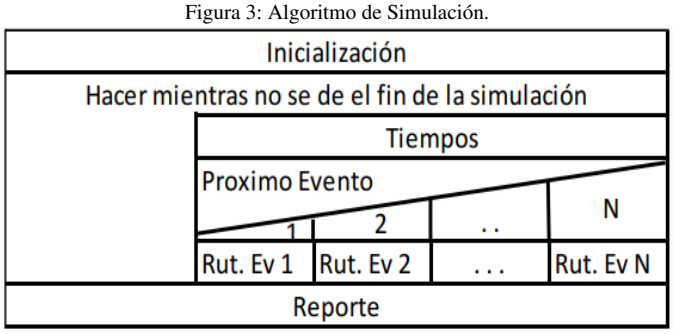
Figura 2: Gráfica de la Q(t)



2. Metodología.

2.1. Algoritmo de Simulación.

Figura 3: Algoritmo de Simulación.



El algoritmo consiste en un bucle de eventos como se ve en la imagen anterior. El fin de este bucle depende de las condiciones finales definidas. En este caso utilizaremos las siguientes:

- Que el reloj no supere las 50 unidades de tiempo.
- Que ya no haya clientes en cola.
- Que el servidor este *disponible*.

Se deben cumplir las 3 condiciones en simultaneo para que termine la ejecución del programa.

2.1.1. Inicialización.

Inicialmente se inicializan las variables que se analizaran/utilizaran en la simulación. Estas son:

- **Tiempo Medio Entre Arribos:** En este caso se define como 7,0.
- **Tiempo Medio de Servicio:**En este caso se define como 9,0.
- **Estado del Servidor:** Se debe inicializar como *disponible*.
- **Tiempo de Servicio Acumulado:** Se debe inicializar en *0*.
- **Demora Acumulada:** Se debe inicializar en *0*.
- **Número de Clientes en Cola:**Se debe inicializar en *0*.
- **Área Q(t):** Se debe inicializar en *0*.
- **Cantidad de Personas que Entraron al Servidor:** Se debe inicializar en *0*.
- **Primer Evento:** Se genera el tiempo del primer evento con la función exponencial denominada en la sección 1.2 con $\lambda = 7{,}0$ (*Tiempo Medio Entre Arribos*).

Siempre se debe realizar la codificación de manera que el primer tiempo generado sea un *Arribo*.

2.1.2. Tiempos.

En este método se analiza si el siguiente evento será una *partida* o un *arribo*.

2.1.3. Próximo Evento.

Analiza según el tipo de evento que sea la acción que va a tomar.
Los cuatro casos posibles son los siguientes:

- Suponiendo que el evento sea un arribo, primero analiza la disponibilidad del servidor. En el caso que este esté *ocupado*, se debe agregar un nuevo cliente a la cola. Finalmente, se calcula y suma el *Área Q(t)* correspondiente.
- De lo contrario, el cliente que llega e ingresa directamente al servidor, el cual cambia de estado a *ocupado*. Se debe calcular el tiempo de servicio que este va a ocupar, utilizando la distribución exponencial con $\lambda = 9{,}0$ y se suma el tiempo que demorará el servicio al *Tiempo de Servicio Acumulado*. También se debe aumentar la cantidad de clientes que completaron la demora satisfactoriamente.
- En el caso que sea una partida, se deberá verificar si hay clientes esperando en cola. De haberlos, ingresa un nuevo cliente al servido siguiendo el algoritmo *FIFO (First in First out)*, en el cual se selecciona al cliente que llego primero para que ingrese al servidor. Se calcula el Tiempo de Servicio del nuevo cliente y se lo añade al *Tiempo de Servicio Acumulado*, también se calcula cuanto tiempo estuvo el cliente en la cola y se lo añade a la *Demora Acumulada*.
  Posteriormente, quitamos a ese cliente de la cola de espera por el servidor. Finalmente, se calcula y suma el *Área Q(t)* correspondiente.
- De estar la cola vacía, y si no es el fin del algoritmo, se cambia el estado del servidor a *disponible* y se deberá volver a forzar que el siguiente evento sea un arribo.

2.1.4. Reporte

Se calculan las *Medidas de Rendimiento* de la manera enunciada en la sección 1.4. Utilizando las fórmulas 1.4.1, 1.4.2 y 1.4.3.

2.2. Iteración del algoritmo

El algoritmo debe correrse 1000 veces y para cada iteración del mismo se debe calcular el promedio de las medidas de rendimiento acumuladas hasta el momento.

2.3. Hipótesis

Debemos poder demostrar que el valor promedio obtenido luego de la corrida nro. 1000 se aproxime lo mas posible a los valores generados con la calculadora determinística con iguales parámetros para el *Tiempo Medio Entre Arribos* y *Tiempo Medio de Servicio* que los utilizados en la simulación. Los valores generados por la calculadora son:

- **Número Promedio de Clientes en Cola** = 2.7222.
- **Utilización Promedio de los Servidores** = 0.7778.
- **Demora Promedio por Cliente** = 0.3889.

3. Casos de estudio

Para poder simular este modelo, utilizamos el lenguaje de programación *Python* con la implementación de las librerías *matplotlib* para el dibujo de las gráficas, *numpy* para los cálculos probabilísticos y *array* para el manejo de arreglos. El programa realiza 1000 corridas de la simulación M|M|1 para luego realizar los análisis y gráficas correspondientes.

3.1. Método Inicializar

En este método,se inicializa en cero el reloj que marcará los instantes de tiempo durante la simulación y se establece como *disponible* al servidor. Se guarda el primer arribo en un arreglo llamado *Lista de Eventos* en su primera posición, se fuerza que el primer evento no sea una partida, guardando en la segunda posición del arreglo el valor 9999 y de esa forma, en la primer instancia, sea menor el valor que se guarda representando al arribo.

Los siguientes métodos descriptos se realizaran en un bucle hasta alcanzar las condiciones finales enunciadas en 2.1

3.2. Tiempos

Mediante una comparación se analiza si el valor en la primera posición es menor que el que está en la segunda. En caso de ser afirmativo, se define al evento como un **arribo** y se asigna el valor de la primera posición al reloj. En el otro caso, se define al evento como **partida** y también se le asigna el valor de la segunda posición al reloj.

3.3. Arribos

Debido a que **todo arribo desencadena un nuevo arribo**. sobreescribimos la primera posición del arreglo *ListadeEventos* con un valor que se obtiene mediante la suma del reloj con otro valor exponencial generado.

En el caso de que el servidor este ocupado, obtenemos el área Q(t) desde el momento actual del reloj hacia atrás (TiempoUltimoEvento) de manera que:

AreaQ(t) = AreaQ(t)+ NroDeClientesEnCola * (Reloj -TiempoUltimoEvento))

Guardamos el valor del reloj para saber cuando el cliente llego a la cola y luego, poder calcular la demora. Esto lo guardamos en un arreglo *float* denominado *Cola*.

Si el servidor está disponible, se cambia su estado a **ocupado**, sobreescribimos la segunda posición del arreglo *ListadeEventos*, con el valor obtenido de la suma del reloj y la generación de otro valor exponencial pero esta vez con el tiempo medio de servicio. A esa futura partida le restamos el reloj para poder acumular el tiempo de servicio.Finalmente, sumamos una unidad a la cantidad de clientes que completaron demora.

3.4. Partidas

Primero, se consulta si hay clientes en cola.
En el caso de que no los haya, establecemos al servidor como disponible y forzamos un arribo volviendo a adjudicarle el valor 9999 a la segunda posición del arreglo *ListaDeEventos*.

En el caso de que si haya clientes en cola, generamos la próxima partida (sobreescribiendo la segunda posición del arreglo *ListaDeEventos* con el resultado de la suma del reloj con un valor exponencial generado con el tiempo medio entre servicios).

Acumulamos la demora con el valor actual del reloj menos el valor del reloj cuando el cliente ingresa a la cola, que esta guardado en el arreglo *Cola* en la primera posición.

Actualizamos el contador de clientes que completaron la demora, para luego poder acumular el tiempo de servicio (restandole a la primera posición del arreglo *ListaDeEventos* el reloj)

Luego, calculamos el Área bajo Q(t) del periodo anterior (Reloj - TiempoUltimoEvento) de manera que en los arribos. Disminuimos la cantidad de clientes en cola en 1 y mediante la función *pop*, quitamos el valor actual en la primera posición de el arreglo *Cola* para que el tiempo de ese arribo ya no figure en el arreglo. La función *pop* mueve todos los valores del arreglo hacia la izquierda.

3.5. Reportes.

Para terminar, guardaremos los promedios de tres promedios de medidas de rendimiento calculadas en 3 arreglos para poder luego graficarlas en el método *análitica*.

La fórmula utilizada para obtener las medidas están enunciadas en 1.4.1,1.4.2 y 1.4.3.

3.6. Analítica.

Este método se encarga de graficar los reportes conseguidos durante las 1000 corridas con uso de los métodos de la librería *matplotlib*. Las gráficas obtenidas son las siguientes:

Figura 4: Demora Promedio por Cliente.

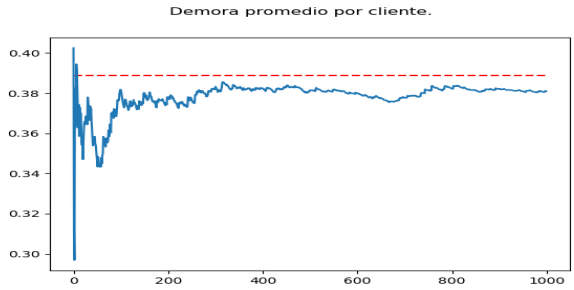


Figura 5: Número Promedio de Clientes en Cola.

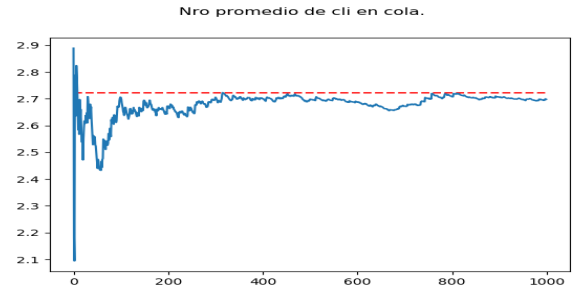
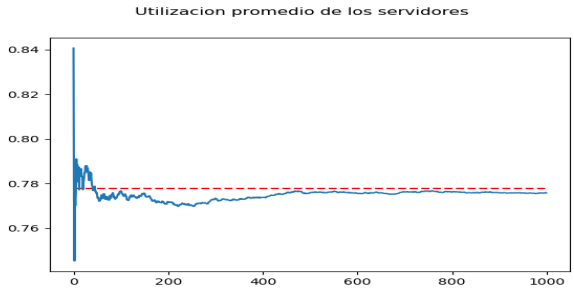


Figura 6: Utilización Promedio del Servidor.



En las tres gráficas la linea punteada roja que las atraviesan son las referencias a los números obtenidos en la calculadora, los cuales están referenciados en la sección 2.3

4. Conclusiones

El valor promedio final obtenido de las 1000 corridas, dan muy aproximados a los obtenidos en la hipótesis en la sección 2.3. Estos son:

- **Número Promedio de Clientes en Cola** = 2.72222223.
- **Utilización Promedio de los Servidores** = 0.77777778
- **Demora Promedio por Cliente** = 0.388888889.

Por esto podemos concluir que la simulación fue exitosa.

4.1. Modificación de tasas.

Al igualar las tasas de tiempo medio de servicio y entre arribos, no se llega a un resultado convergente.esto se debe a que al ser los tiempos iguales, los clientes en cola varían demasiado y no se encuentran similitudes entre las distintas corridas. Tiene relación con el hecho de que se puede ver como ambas funciones en la distribución exponencial convergen a 0 al mismo tiempo y por lo tanto las medidas de rendimiento son totalmente aleatorias.

En el caso que la tasa de servicio sea menor que la de arribos, nos encontramos con el hecho de que no se llegan a cumplir las condiciones finales del algoritmo, ya que siempre hay un nuevo arribo previo a un servicio, y por ende, la cola nunca esta vacía.

5. Apéndice.

El código que corresponde a este trabajo práctico se encuentra en el siguiente link. `https://github.com/orcorsetti/Simulacion/blob/master/MM1.py`