

Práctica 3 SS: Modelos de Simulación Dinámicos y Discretos

José Manuel Pérez Lendínez, 26051613-1

December 20, 2019

Contents

1	Método de incremento fijo de tiempo.	3
2	Método de incremento variable de tiempo.	4

1 Método de incremento fijo de tiempo.

En este caso vamos a realizar un programa que simule la carga de trabajo de un servidor. Los clientes llegan al servidor y esperan en una cola fifo hasta que este quede libre para ser utilizado. Tanto el momento en el que llegan los clientes como el tiempo que necesitan utilizar el servidor vienen dado por dos generadores de variables aleatorias independientes e idénticamente distribuidas. Estos generadores nos devolverán un valor que determinara un momento de tiempo que pasara hasta la llegada de un nuevo cliente o el tiempo de utilización del servidor. De esta manera le sumaremos este valor el tiempo actual de nuestro simulador para obtener los datos que nos interesa.

El programa recibe 4 valores como entrada con el siguiente formado.

./simulador TiempoDeLlegada N°Clientes TiempoDeServicio N°DeRepeticiones

Las variables indicaran lo siguiente:

1. **Tiempo de llegada (tlleg):** Sera el valor utilizado para generar los tiempos de llegadas al servidor. De forma que si le pasamos un 1 equivaldría a 60 minutos y en caso de pasarle por ejemplo 0.15 serian 9 minutos.
2. **N° de clientes:** El programa para la ejecucion cuando atienda a un numero exacto de clientes, que indicaremos con este parámetro.
3. **Tiempo de servicio(tserv):** Utiliza los mismos tipos de valores que la variable para tiempos de llegada. Pero en este caso indicaremos el tiempo que un cliente utiliza el servidor.
4. **N° de repeticiones:** La simulación la repetiremos un cierto numero de veces para obtener la media de los resultados obtenidos. Ese numero de simulaciones sera indicado por este parámetro.

El simulador nos devolverá una media de tiempo ocioso del servidor y la media de clientes en cola para utilizar el servidor.

El tiempo en la simulación incrementara de forma fija en una unidad. Vamos a realizar realizar varias mediciones y mostrarlas en la siguiente tabla para ver como se comporta el programa para valores distintos en los parámetros de tiempo de servicio y llegada.

Para estos datos he utilizada 10000 clientes y lo he repetido 50 veces para sacar las medias. Para la tabla vamos a utilizar las medidas de las medidas nombradas en la practica horas, medias horas, minutos, segundos,decimas de segundos y milisegundos.

Para trabar con horas tlleg = 0.15 y tserv = 0.1, si se quiera por ejemplo trabajar con medias horas solo tendríamos que multiplicar estos por 2, tlleg = 0.30 y tserv = 0.2. Otro ejemplo seria trabajar con segundos que tendríamos que multiplicar 0,15 y 0.1 por 3600.

T. de llegada (tlleg)	T. servicio(tserv)	Media de clientes en cola	Media % de tiempo ocioso Servidor	T. medio de ejecución (seg)
0.15	0.1	0.0233562	0.135782	0.00116736
0.30	0.2	0.215516	2.98828	0.00159546
9	6	1.26906	31.3571	0.00200088
540	360	1.34331	33.246	0.0173753
5400	3600	1.30638	33.5554	0.150701
54000	36000	1.35409	33.237	1.37355

En la tabla se ve claramente como los valores para las media de cliente en cola y tiempo ocioso del servidor llega un momento en que prácticamente y pasan de ser muy bajas a ser siempre parecida. Esto se debe a que cuando se trabajan con unidades de tiempo altos como pueden ser las horas (tlleg = 0.15 y tserver = 0.1) o medias horas (tlleg = 0.3 y tserver = 0.2) los generadores obtienen valores muy bajos, siendo casi 0. Esto hace devuelva 1 casi siempre para el tiempo de servicio y llegada, los clientes estén llegando continuamente y el servidor este ocupado prácticamente siempre. En cambio al empezar a utilizar tiempos mas pequeños, a partir de minutos (tlleg = 9 y tserver = 6), si se tiene mas en cuenta la aleatoriedad y deja de devolver solo valores próximos a 0 consiguiendo ya resultados mas realistas.

Una vez visto esto y sabiendo que con medidas de tiempo mas pequeñas (tlleg y tserv mas grandes) analizaremos en estos el tiempo medio de cola y ocioso del servidor. Con esto vemos claramente como el servidor se mantiene ocioso prácticamente un 33% del tiempo y en cola tenemos una media de 1.3 clientes.

A la hora de analizar el tiempo se observa como va creciendo, esto se debe a que con unidades de tiempo mas pequeñas hace falta incrementar mas veces el reloj del simulador para llegar a un evento.

2 Método de incremento variable de tiempo.

El método de incremento variable se basa en el condigo anterior añadiendo dos cambios sencillos. Por esto la forma del llamar al programa es la misma que en el caso anterior.

Los cambios principales se basa en que dejamos de trabajar con el tiempo en como entero y utilizaremos un float. Nuestro generadores anteriores tenían que redondear el float para devolver un numero entero, en este método esto no es necesario por lo que el generador devolverla el float sin la necesidad de redondear ni comprobar si el redondeo lo convirtió en un 0. El ultimo cambio

es que el reloj no aumenta de unidad en unidad, el reloj en este caso avanza hasta el evento mas cercano en el tiempo. En nuestro caso sera o la llegada de un nuevo cliente al servidor o que un cliente deje de utilizar el servido.

Vamos a ejecutar con los mismos datos que en el ejemplo anterior para comparar.

T. de llegada (tlleg)	T. servicio(tserv)	Media de clientes en cola	Media % de tiempo ocioso Servidor	T. medio de ejecución (seg)
0.15	0.1	1.32145	33.396	0.0009926
0.30	0.2	1.34731	33.1729	0.00103374
9	6	1.33189	33.3837	0.00105938
540	360	1.36657	33.4256	0.00105644
5400	3600	1.33417	33.0666	0.0010442
54000	36000	1.32711	33.2303	0.00104676

Empecemos analizando el tiempo. En este caso el tiempo no aumenta solo una unidad, sino que salta al evento mas cercano. Esto hace que el tiempo de ejecucion no dependa de la distancia entre un evento y el estado actual, si no del numero de eventos que se den durante la ejecucion del simulador. Para este ejemplo he ejecutado para 10000 cliente, y cada cliente utiliza el servidor una vez. Solo tendremos 20000 eventos y seran lo que necesitemos procesar. Da igual la unidad que utilicemos, solo se tiene el cuenta la cantidad de evento que tendremos que ejecutar. Por tanto el tiempo medio de ejecucion como se ve en la tabla es el mismo para todos los ejemplos quitando las pequeñas sobrecargas que pudiera tener el procesador al ejecutarlos.

Si nos centramos en las medias de clientes en cola y tiempo ocioso del servidor, vemos que no ocurre lo mismo que con el incremento fijo. En este caso no pasamos de unas medias muy bajas a unas mas altas, sino que se mantienen las medias durante todas las ejecuciones en los mismos valores. Esto se debe a que en este caso como explique anteriormente al estar trabajando con float no tenemos el problema con unidades de tiempo grandes, como por ejemplo horas ($tlleg = 0.15$ y $tserv = 0.1$). Aunque obtengamos valores próximos a 0 con los generadores, nuestro simulador si puede trabajar con estos y no los redondea a 1 como en el apartado anterior. Con esto podemos asegurar que con el incremento variables, la unidad de tiempo que elijamos no nos afectara a la hora de obtener las medias.

Vamos a comparar nuestros resultados con los resultados obtenidos teóricamente. Cuando $n \rightarrow +\infty$, sabes teóricamente que:

$$\rho = \frac{t_{serv}}{t_{IIeg}}, \quad Q(n) \rightarrow \frac{\rho^2}{1-\rho}, \quad PTO(n) \rightarrow 100 * (1 - \rho)$$

Vamos a sustituir y resolver con los valores para horas:

$$\rho = \frac{0.1}{0.15} = 0.667$$

$$Q(n) \rightarrow \frac{0.667^2}{1 - 0.667} = 1.334$$

$$PTO(n) \rightarrow 100 * (1 - 0.667) = 33,334$$

Con esto vemos como teóricamente tenemos resultados prácticamente iguales a los conseguidos con el incremento variable. En cambio si miramos los resultados que conseguimos con el incremento fijo:

T. de llegada (tl-leg)	T. servicio(tserv)	Media de clientes en cola	Media % de tiempo ocioso Servidor	T. medio de ejecución (seg)
0.15	0.1	0.0233562	0.135782	0.00116736

Vemos que no se acercan ni lo mas mínimo a los resultados teóricos. Esto ya nos asegura todo los indicios anteriores de que el metodo fijo tiene con valores altos de tiempo.