Simulación
Teoría de Colas

Rodríguez Cuervo, Juan David.

Abstract—En el presente trabajo se estudiará un sistema de filas, en donde los tiempos de llegada de los clientes al sistema, así como de los servidores, son descritos con variables aleatorias. Al describir una variable aleatoria bajo una distribución determinada, se puede modelar el sistema en cuestión, pudiendo establecer los valores óptimos de recursos, en este caso cantidad de servidores, que se requieren para evitar las filas sigan creciendo indefinidamente. A partir de todo el modelo matemático del sistema se establecerán los valores mínimos requeridos para que el sistema permanezca estable en toda su corrida.

Keywords—Computational modeling, Data visualization, Model checking, Stochastic systems, Systems simulation.

1 Introducción

En la teoría de colas (Queueing Theory), se pretende entender el comportamiento de un sistema de servicio. En otras palabras, aquellos sistemas en los que entran clientes para ser atendidos por un servidor. A medida que van llegando más clientes al sistema, si el servidor se encuentra ocupado (en servicio dedicado a otro cliente), se debe realizar una espera hasta que el servidor pueda atender a los clientes entrantes. En general, el concepto de servicio que se trabaja para cada servidor es FIFO (First in, first out).

Dependiendo del balance que tengan las distribuciones de probabilidad de la llegada de los clientes al sistema, y del servicio de cada servidor, puede que las filas (esperas) comiencen a presentar inestabilidad (tiendan a la divergencia). Para evitar esto, el sistema de queueing es simulado, variando los parámetros de servicio, de manera que se busquen aquellos valores que minimicen los recursos y que permitan al sistema converger en su servicio (se minimicen tiempos de espera, longitud de las colas).

En el presente trabajo se va a tomar un caso de estudio particular, que va a ser estudiado para observar el comportamiento que presente. De este modo, se pretenderá responder a la cuestión de cuál es la cantidad óptima de servidores necesarios para estabilizar el sistema.

Este modelo será implementado en código, con el lenguage de Python. Al final, se pretenderá mostrar qué parámetros del sistema son suficientes para asegurar la estabilidad del mismo, y se demostrará su validez. Para ello, se describirán algunas características del sistema, y se aplicarán herramientas de visualización de datos que ilustren la validez de los parámetros seleccionados.

2 Caso de estudio

El sistema que se propone como objeto de estudio, en principio, consta de dos servicios en línea. Esto significa que un cliente, al entrar en el sistema, es atendido por un servidor, y al salir de este, entra a un segundo servidor.

Los clientes llegan al sistema en tiempos aleatorios, con una distribución exponencial, cuya media es 0.5 clientes cada minuto. El primer servidor, por su parte, tiene una distribución de servicio uniforme, variando sus tiempos entre 0.5 y 1.5 minutos. Por último, los servidores que se encuentren atendiendo el segundo servicio, tendrán distribuciones de servicio uniformes entre 2 y 2.5 minutos.

Como se observar en la imagen 1, el sistema, definido con un sólo servidor en el segundo servicio, no es estable. Esto significa que su cola se incrementará indefinidamente con el paso del tiempo. Como se planteó previamente, este es el caso que se busca evitar. De este modo, la estrategia de cambio será incrementar el número de servidores para el segundo servicio, de manera que el sistema llegue a un estado estable. El objetivo es la optimización, por lo que se buscará el mínimo número de servidores tal que el sistema se estabilice.

3 Modelo de Servidores

3.1 Los Servidores como objetos

En el sistema computacional los servidores son tratados como objetos. Se considera un servidor aquel elemento del sistema que realiza alguna acción dentro de una distribución particular de tiempos.

Por tanto, lo importante en este modelo de colas con relación a los servidores, es saber la cantidad de actividades a realizar y cuál es el tiempo simulado para cada una.

Siguiendo este concepto, se puede observar con claridad que la llegada de los clientes al sistema puede ser modelado como un servidor.

En el código se observará un archivo denominado ”Server.py”. Este archivo contiene la definición del objeto Servidor o Server dentro del sistema. En las primeras líneas del archivo principal (”fila.py”), se puede observar la definición y tratamiento de los objetos Servidores. Para

crear un servidor, se define una función de distribución aleatoria, ya sea constante, uniforme, exponencial, o cualquier otra que se le defina. Esta función de distribución es la que será aplicada a valores aleatorios entre 0 y 1, de manera que el objeto genere una lista de números aleatorios que siguen la distribución especificada.

Así, en la creación de las instancias del objeto Server, se define la función de distribución para el servidor. Pero luego, cuando ya se requiera la lista de datos en concreto, se le envía como parámetro la cantidad de valores a generar al servidor, y con la función definida internamente desde la definición, se genera la cantidad de valores solicitados.

3.2 Distribuciones

En la simulación de los comportamientos aleatorios, generalmente las funciones generadoras presentan ciertas distribuciones particulares. En el sistema presente se asumen estas características, esbozadas en el Caso de Estudio.

A continuación se van a describir brevemente las distribuciones relevantes para este trabajo y, a *grosso modo*, su aplicación matemática dentro de los componentes del sistema.

Par cada distribución, se tendrán dos variables en común, a decir, x y r. La primera se refiere a la variable aleatoria con la distribución que se quiere obtener, mientras que r es una variable aleatoria (por lo general la estándar generada por el computador) uniforme con dominio $[0, 1]$.

3.2.1 Distribución uniforme

Se denota como $U(a, b)$, en donde $a < b \in \mathbb{R}$. Después de todo el proceso de deducción matemática, se obtiene que:

$$r = \frac{x - a}{b - a}$$

De esta expresión, que depende de x, se puede obtener fácilmente que:

$$x = r(b - a) + a \tag{1}$$

Cuando en el sistema se haga referencia a que un servidor tiene una distribución uniforme, por tanto, significa que se usará (1) para, dado un aleatorio r entre 0 y 1, generar un valor aleatorio con distribución uniforme entre a y b.

3.2.2 Distribución exponencial

De forma análoga al caso anterior, se tiene una función acumulada de probabilidad exponencial (exponencial negativa, para ser más precisos), de donde, si se integra en su dominio, se obtiene:

$$r = 1 - e^{-\lambda x}$$

De esta expresión, despejando x:

$$x = \frac{-\ln(1 - r)}{\lambda} \tag{2}$$

3.3 Generación de datos

Para la generación de datos, se aplica el concepto de los servidores como objetos que se mencionó previamente.

Después de definir el servidor con su función de distribución, se generan los valores d tiempos. Hay una generación de tiempos para los clientes que entran, y para cada uno de los servidores que atienden.

En la Fig. 1 se observa el comportamiento de la cola con el modelo original de un solo servidor para el segundo servicio.

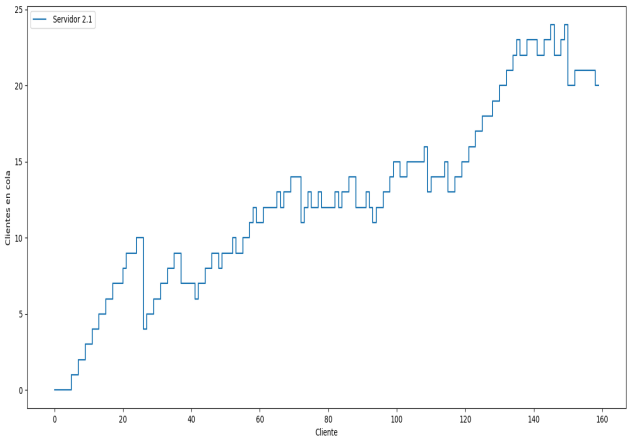

Fig. 1: Modelo original

4 Modelo del Sistema

En esta sección se pretende resumir las fórmulas bajo las que se estructuró el sistema de simulación. Cada variable a trabajar es una característica del sistema, en particular, con respecto a un cliente i. Por tal razón, cada una ha de considerarse como una lista de valores que muestran el comportamiento en sistema especificado, paso a paso.

Para las siguientes expresiones, el tiempo será abreviado con t. De igual modo, momento de llegada será abreviado con m, los inicios con ini, y las terminaciones con $term$. El servicio será denotado por $serv$.

El primer servicio consta de un solo servidor, mientras que el segundo servicio, la cantidad de servidores es variable. Por esta razón, las expresiones correspondientes al segundo servicio incluirán, en su mayoría, un subíndice k que representa el servidor del que se esté refiriendo.

4.1 Primer Servidor

$$
\begin{aligned}
m_i &= t\_llegada_i + m_{i-1} \\
ini\_serv_i &= \max(m_i,\, term\_serv_{i-1}) \\
term\_serv\_1_i &= ini\_serv_i + duracion\_serv\_1_i \\
t\_sistema &= term\_serv\_1 - m \\
t\_espera &= ini\_serv\_1 - m \\
queue\_1_i &= |\{x \in term\_serv\_1[0:i] : x > m_i\}|
\end{aligned}
$$

4.2 Segundo Servicio

Antes de comenzar con las ecuaciones, conviene definir algunos puntos previos. En particular, se requiere conocer, para cada cliente en la fila, cuál es el servidor del segundo servicio que primero podría atenderlo. Esto es, iterando sobre los servidores, se determina como óptimo aquel servidor cuyo tiempo de finalización de servicio actual, sea menor al de los otros. Esto es, que se desocupa primero. A este servidor óptimo, se le va a denotar como $serv\_opt_i$, para cada cliente i.

$$
\begin{aligned}
ini\_serv\_2_{k,0} &= \begin{cases} term\_serv\_1_0, & k = 0 \\ 0, & k \neq 0 \end{cases} \\
term\_serv\_2_{k,0} &= ini\_serv\_2_{k,0} + duracion\_serv\_2_{k,0}
\end{aligned}
$$

$$
\begin{aligned}
\textbf{si} \quad \mathbf{k} &= \mathbf{serv\_opt_i}: \\
ini\_serv\_2_{k,i} &= \max(term\_serv\_1_i,\, term\_serv\_2_{k,i-1}) \\
term\_serv\_2_{k,i} &= ini\_serv\_2_{k,i} + duracion\_serv\_2_{k,i}
\end{aligned}
$$

$$
\begin{aligned}
\textbf{si} \quad \mathbf{k} &\neq \mathbf{serv\_opt_i}: \\
ini\_serv\_2_{k,i} &= ini\_serv\_2_{k,i-1} \\
term\_serv\_2_{k,i} &= term\_serv\_2_{k,i-1}
\end{aligned}
$$

$$
queue\_2_{k,i} = |\{x \in term\_serv\_2_k[0:i] : x > term\_serv\_1_i\}|
$$

5 Resultados

El requisito del sistema es mantener estabilidad durante 160 clientes. No obstante, como se puede observar en la Fig. 2, se realizó la prueba con 500 clientes, mostrando estabilidad.

El valor máximo de personas en la fila fue de 4. Con sólo 2 servidores se pudo asegurar la estabilidad del sistema, cosa que no se lograba con uno solo.

6 Conclusiones

En el presente trabajo se pudo estudiar un sistema de colas, evidenciando el impacto que tienen las distribuciones de probabilidad de las variables aleatorias del sistema sobre su comportamiento. A medida que los servidores eran agregados o eliminados del sistema, la tendencia del sistema (longitud de la fila) podía comenzar a divergir o a conservar estabilidad en un rango determinado.

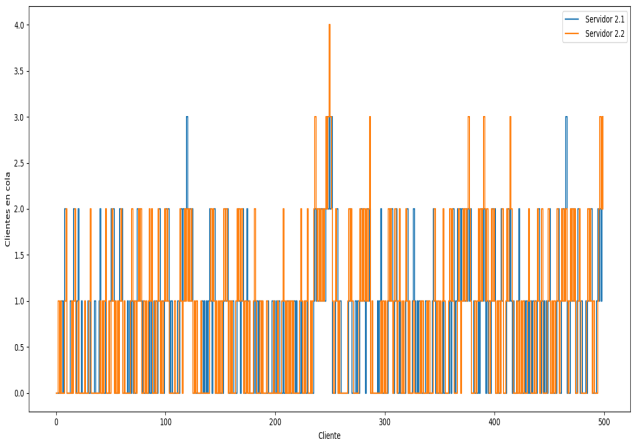

Fig. 2: Modelo con 2 servidores

Por otro lado, se pudo aplicar el concepto de optimización de los recursos, puesto que a medida que se van agregando servidores al sistema, eso significa mayor cantidad de recursos gastados y posiblemente mayor cantidad de tiempo ocioso en los servidores. Así, lo que se buscó fue obtener la configuración mínima suficiente para mantener el sistema estable, de manera que el tiempo ocioso de los servidores fuese mínimo.

De igual modo se establecieron las relaciones internas de un sistema de queueing, observando cómo las variables se afectaban entre sí. Esto se podía evidenciar en que el valor de una variable en un punto del sistema particular, podía depender de sus valores previos o de otros indicadores del sistema.

References

[1] Banks, J., J. Carson, B. Nelson, and D. Nicol, *Discrete-Event System Simulation*, Prentice-Hall, fourth edition, 2004.

[2] Law, A. *Simulation Modeling and Analysis*. McGraw Hill, fifth edition, 2013. Full text articles from Winter Simulation Conference, www.informs-cs.org