Informe del
Primer Proyecto de Simulación

Tema 5

Marco Antonio Ochil Trujillo
Jan Carlos Pérez González

26 de febrero de 2024

Introducción:

El problema se describe de la siguiente forma; los clientes llegan a un sistema que tiene n servidores con retroceso, y las llegadas distribuye M. Cada cliente que llega debe ser atendido primero por el servidor 1 y, al completar el servicio en el servidor 1, el cliente pasa al servidor 2. Cuando un cliente llega, entra en servicio con el servidor 1 si ese servidor está libre, o se une a la cola del servidor 1 en caso contrario. De manera similar, cuando el cliente completa el servicio en el servidor 1, entra en servicio con el servidor 2 si ese servidor está libre, o se une a su cola y así sucesivamente. Después de ser atendido en el servidor n, el cliente abandona el sistema.

La resolución de problemas de este tipo tiene como objetivo mostrar resultados para distintos casos y parámetros para así llevarlo a problemas de la vida real y poder tomar decisiones informadas.

Las variables que describen el problema son:

* Cantidad de Servidores, entero mayor que 1.
* Tiempo de la Simulación, flotante mayor que 0.
* Probabilidad de cada servidor cambiar de un servidor i a un servidor j.

Estos son los datos principales que se necesitan para llevar a cabo la simulación, en la siguiente sección veremos que se agregaron parámetros a la simulación para obtener resultados más ricos.

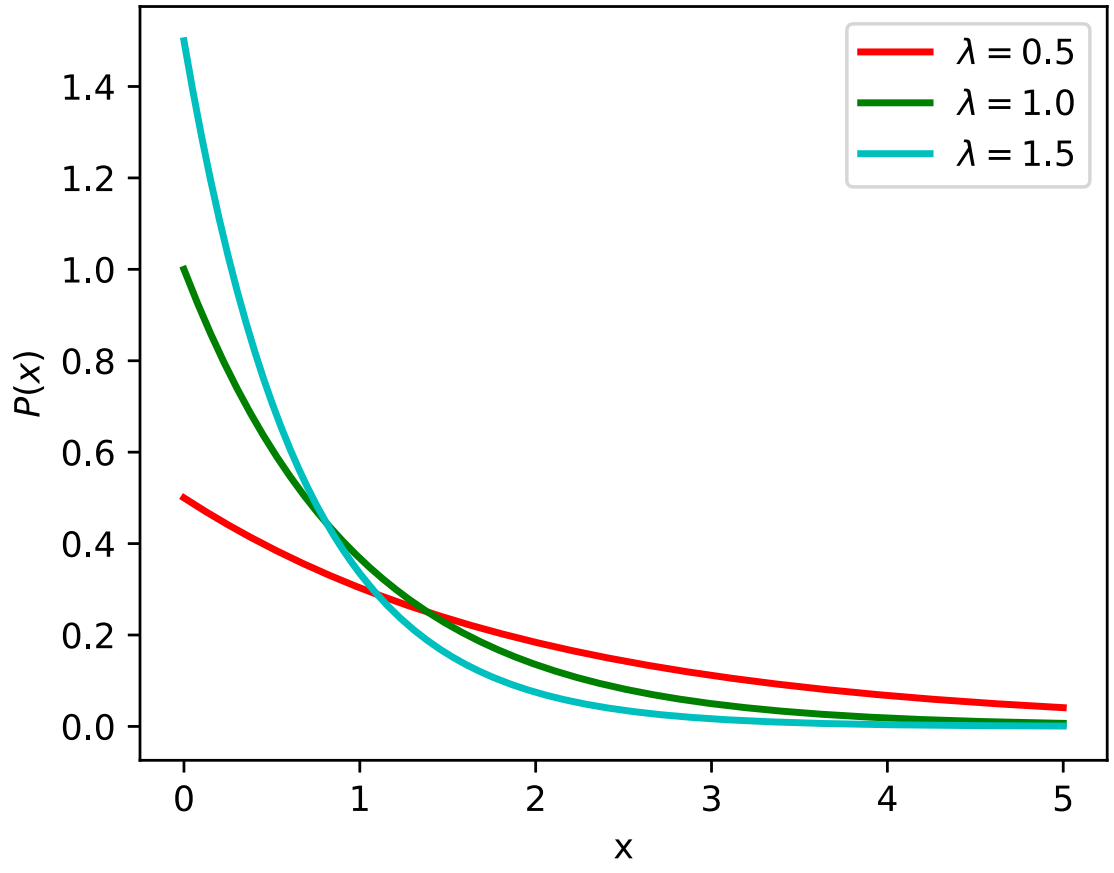
Enriquecimiento del problema:

Para obtener más información en los resultados se modeló al cliente con ciertas propiedades “realistas”, como paciencia, tiempo de espera y cantidad de clientes que está dispuesto a esperar, estas son variables aleatorias uniforme entre 0 y 3, 10 y 50, 0 y 10 respectivamente, la paciencia indica cuántas veces está dispuesto a un cliente a “resistir” que se le mueva de un servidor i a otro j, tal que

j<i, el tiempo de espera es básicamente la cantidad de tiempo que puede esperar estando en la cola y la tercera propiedad es cuántos clientes pueden haber por delante del cliente en cuestión en la cola. Además se modeló el problema usando para cada servidor una distribución aleatoria entre seis distribuciones posibles (exponential, normal, gumbel, laplace, logistic, rayleigh). Además la probabilidad de cada servidor se tomó entre 0.5 y 1 para imprimir algo de dinamismo y que no fuera tan frecuente que un cliente cambie de servidor, pues realentizaría la simulación y no es relevante para un caso de estudio general.

Antes de pasar a la siguiente sección hagamos un análisis simple de los variables aleatorias que se usaron:

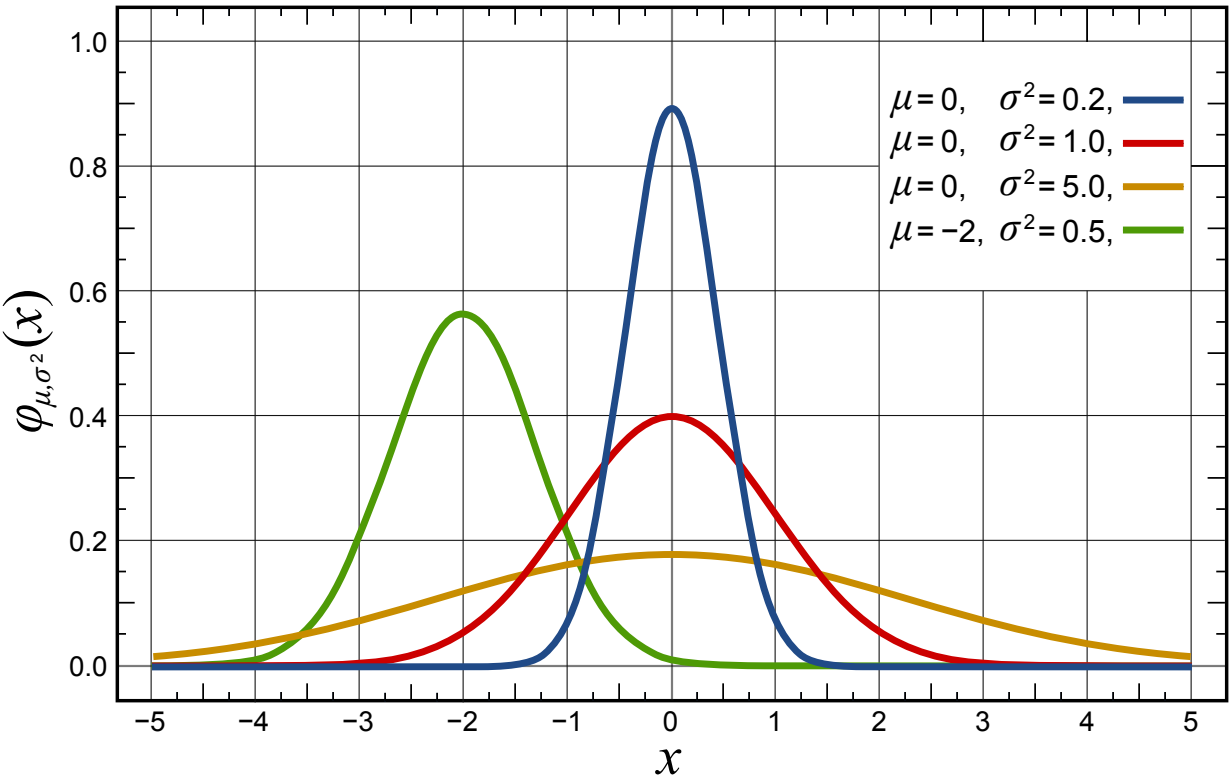
Exponencial:



$E[X] = 1/\lambda \qquad V[X] = 1/\lambda^2$

El valor predefinido del parámetro scale en la función exponencial es 1.0 y representa un valor $\beta = 1/\lambda$, luego $E[X] = 1.0$ y como $\lambda = 1.0$, entonces $V[X] = 1.0$ y $\sigma = 1.0$, aunque en los resultados del problema la media parece mayor porque en realidad lo que se está haciendo es sumando una exponencial cada vez que un cliente entra al servidor, o sea, si entran 5 clientes se suman 5 exponenciales, por eso los valores serán lejanos a 1.

Normal:



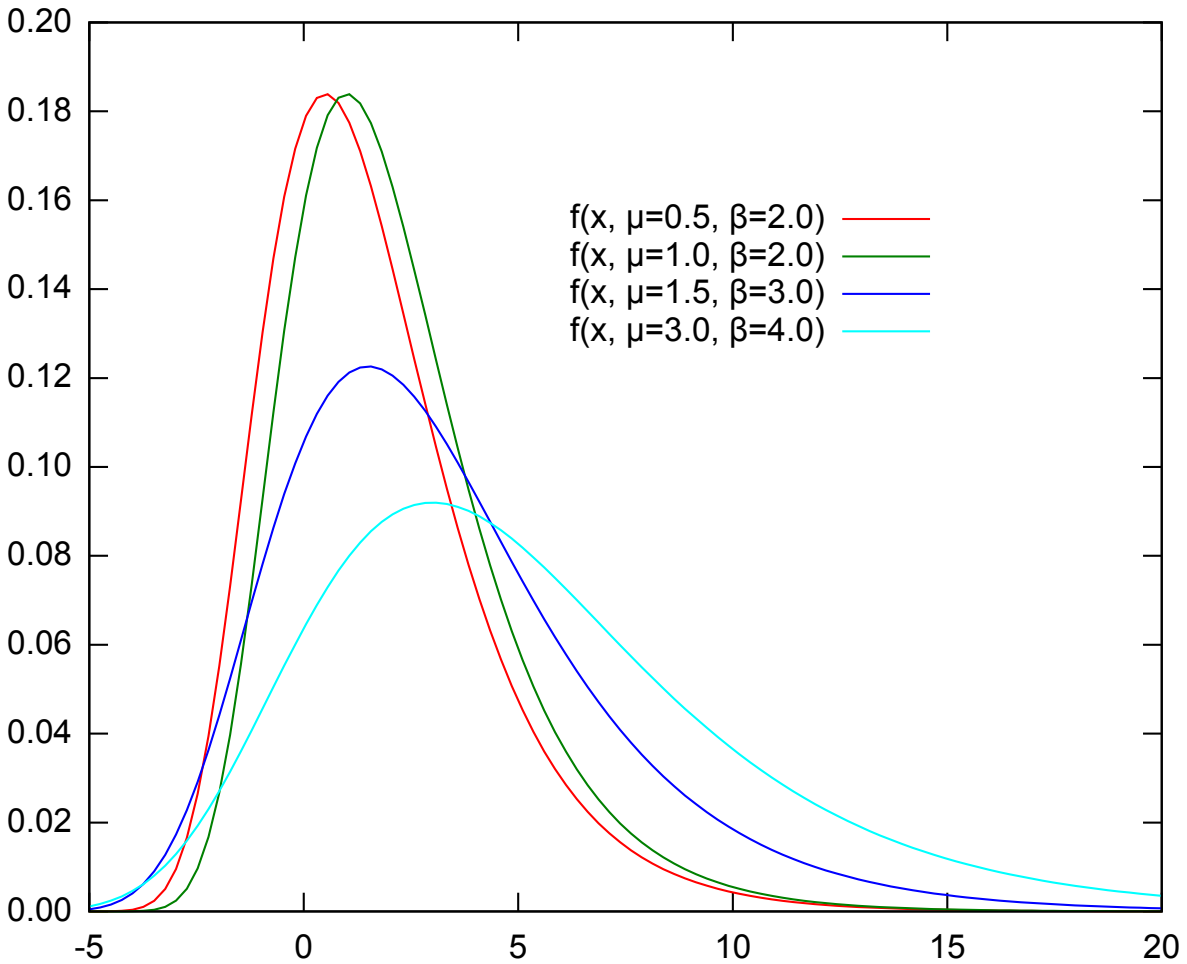
Los valores predefinidos de la función normal son loc = 0.0 y scale = 1.0, lo que representa una función normal con $E[X] = 0.0$ y $\sigma = 1.0$, que puede tomar valores negativos, caso que no tendría sentido en el problema actual, por lo que decidimos utilizar el módulo de dicha función con lo que la probabilidad de

ocurrencia de cualquier valor positivo se duplico, dado que esta función era simétrica a la recta x = 0.

Estas variables aleatorias fueron impartidas en clase a diferencia de las siguientes, las cuales fueron encontradas dentro de las generadas en `numpy.random` y se usaron porque cumplían con las facilidades que brindaban las anteriores, que son: retornar float y no recibir parámetros necesariamente.

Gumbel:

$$f(x;\mu,\beta) = \frac{e^{-(x-\mu)/\beta}}{\beta} * e^{-e^{-(x-\mu)/\beta}}$$



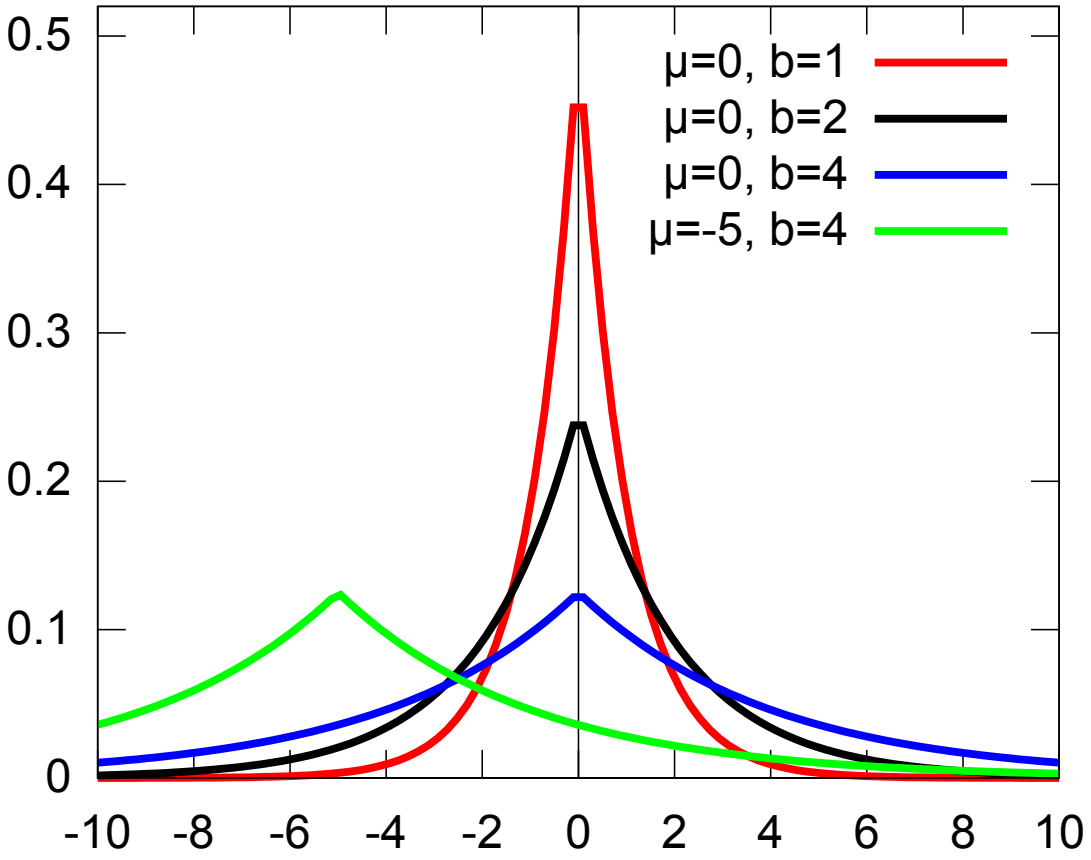
Los valores predefinidos de la función gumbel son loc = 0.0 y scale = 1.0, donde μ = 0.0 y β = 1.0, por lo que la función quedaría:

Con E[X] = μ + 0.57721 * β = 0.57721 y V[X] = = 1.64493, luego σ = 1.28254

Para esta función también fue necesario usar módulo.

Laplace:

$$f(x;\mu,\lambda) = \frac{1}{2\lambda} exp(-\frac{|x-\mu|}{\lambda})$$



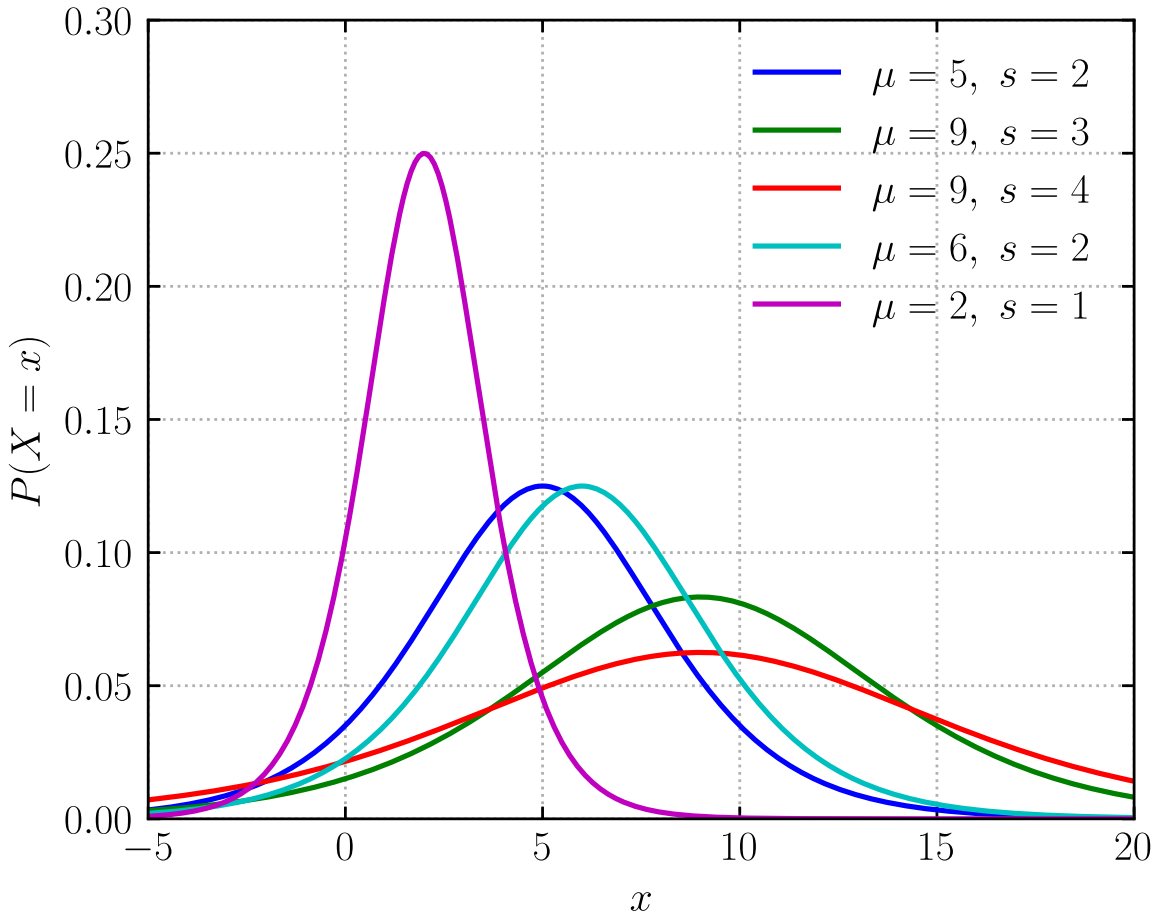
Los valores predefinidos de la función laplace son loc = 0.0 y scale = 1.0, donde μ = 0.0 y λ = 1.0, por lo que la función quedaría:

Con E[X] = μ = 0 y V[X] = 2 * λ^2 = 2, luego σ = 1.41421

Para esta función también fue necesario usar módulo.

Logistic:

$$f(x, \mu, s) = \frac{e^{-(x-\mu)/s}}{s(1 + e^{-(x-\mu)/s})^2}$$



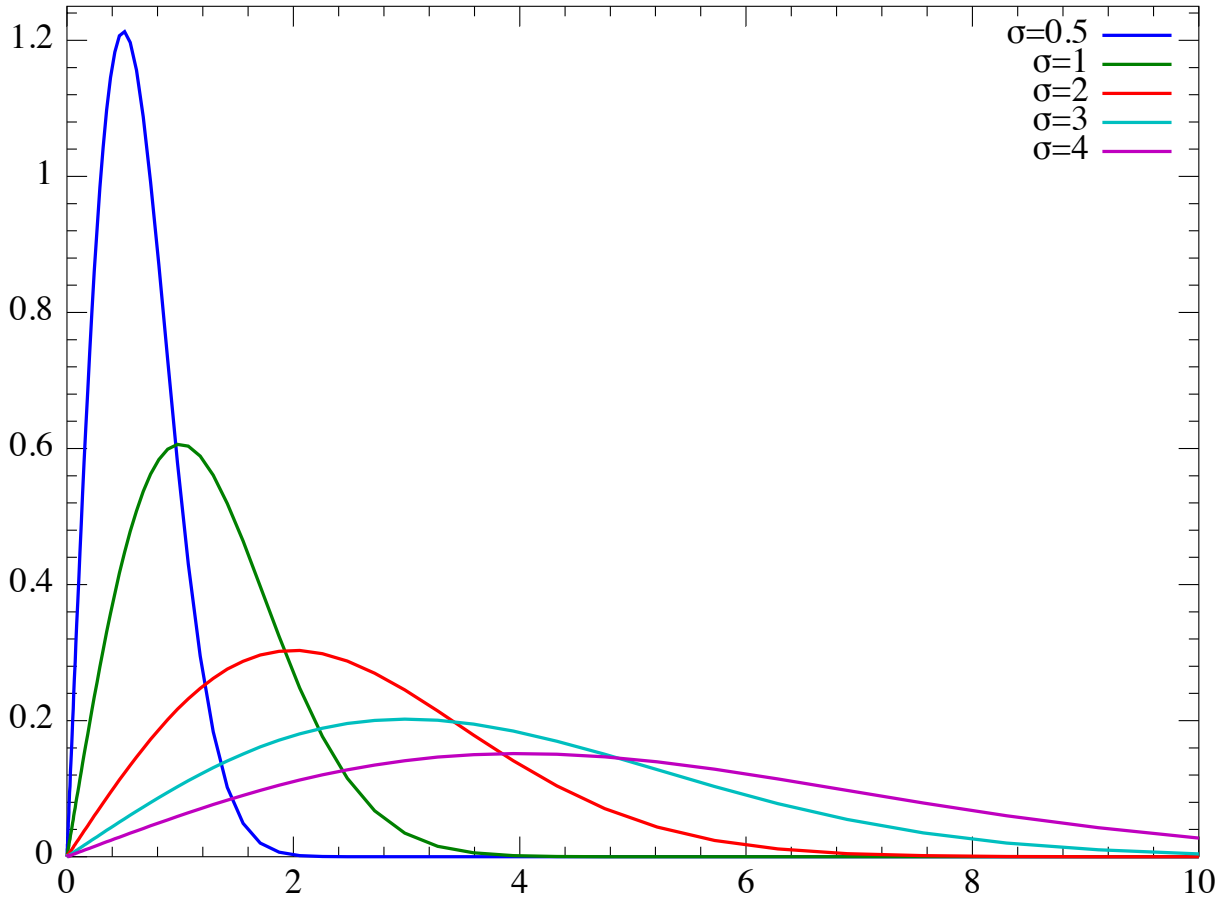
Los valores predefinidos de la función logistic son loc = 0.0 y scale = 1.0, donde μ = 0.0 y s = 1.0, por lo que la función quedaría:

Con E[X] = μ = 0 y V[X] = = 3.28986, luego σ = 1.81379

Para esta función también fue necesario usar módulo.

Rayleigh:

$$f(x; scale) = \frac{x}{scale^2} e^{\frac{-x^2}{2scale^2}}$$



El valor predefinido de la función rayleigh en scale es 1.0, por lo que la función quedaría:

Con E[X] = = 1.25331 y V[X] = = 0.4292, luego σ = 0.65513

Detalles de implementación[1]:

Se utilizaron para el proyecto las bibliotecas `math`, `random` y `numpy`. En `numpy.random` se encuentran las funciones que se van a usar para las distribuciones aleatorias de las que se habló antes.

Para ejecutar la simulación hay que cambiar los siguientes parámetros según se requiera para analizar cada caso:

```
242
243    time_opened = 500
244    servers = 5
245    time_less, stats_finish_mode, stats_server_time, stats_server_lose = simulation(1000, time_opened, servers, True, True, False)
246
```

La función `simulation` recibe la cantidad de simulaciones que se quiere hacer, el tiempo que estará funcionando el servidor, la cantidad de servidores, y tres boleamos que significan si se quiere tener en cuenta algunas de las propiedades de los clientes que se mencionaron anteriormente (en el orden en el que se mencionaron).

1: El código se encuentra debidamente comentado en el repositorio en caso de que algo no quede lo suficientemente claro.

Esta función devuelve:

i) `time_less`: tiempo extra que el servidor está trabajando porque hay clientes en la cola

ii) `stats_finish_mode`: es una lista que contiene cinco listas donde cada una posee:

ii.i) Cantidad de personas que entran al sistema

ii.ii) Cantidad de personas que completan el recorrido del sistema

ii.iii) Cantidad de personas que se van del sistema por esperar mucho tiempo

ii.iv) Cantidad de personas que se van del sistema por regresas muchas veces

ii.v) Cantidad de personas que se van del sistema por cola larga.

iii) `stats_server_time`: Para cada variable aleatoria cuanto se demoran las personas en un servidor con dicha variable como generador de tiempo

iv) `stats_server_lose`: Para cada variable aleatoria cuantos clientes pierde.

Para cada uno de estos resultados se halla su media y su desviación estándar, pero acerca de estos resultados se abordará en profundidad en la siguiente sección.

Para controlar el flujo de los clientes están los métodos

`arrive:` Cuando llega un cliente nuevo se agrega y se determina si se añade al servidor 1 (si es que está "dispuesto a hacer la cola"), además se actualizan los valores del tiempo de arribo y de la cantidad de arribos. El tiempo de arribo guarda cuándo será el próximo arribo, el cual distribuye exponencial.

`exit_server:` Analiza cada criterio de salida del sistema, explicados anteriormente y actualiza los valores correspondientes.

`simulate:` Es el método que controla el flujo como tal, ya que aquí se hacen todas las simulaciones necesarias y se van guardando los resultados.

`time_wait_review:` Revisar los clientes para eliminar los que se cansaron de esperar y abandonaron sus colas.

Resultados y conclusiones:

Hacemos notar que las ocho filas que contienen las tablas es el resultado de activar y desactivar cada una de las restricciones que se implementaron de manera extra para enriquecer(paciencia, tiempo de espera, cantidad de personas a esperar). Las siguientes tablas y conclusiones son para un tiempo igual a 100 y 5 servidores.

Tiempo Extra	Clientes Recibidos	Clientes Atendidos	Perdida por Espera	Perdida por Retroceso	Perdida por Cola
81.9 // 24.9	100.5 // 9.9	100.5 // 9.9	X	X	X
39.7 // 9.9	100.2 // 10.1	56.1 // 7.7	44.1 // 12.1	X	X
63.4 // 24.0	99.8 // 10.2	79.7 // 9.0	X	20.1 // 4.4	X
48.2 // 14.6	100.4 // 10.1	46.0 // 5.3	X	X	54.4 // 10.1
29.7 // 7.5	100.7 // 9.9	59.5 // 7.8	31.1 // 11.8	10.2 // 3.3	X
27.7 // 9.3	100.3 // 10.3	45.2 // 6.0	13.0 // 4.5	X	42.1 // 8.8
22.1 // 7.5	99.6 // 10.0	42.5 // 5.1	X	18.0 // 3.9	39.2 // 9.0
16.3 // 5.1	100.3 // 10.0	48.1 // 5.9	7.4 // 3.6	10.2 // 3.0	34.6 // 8.4

Conclusiones:

- Añadir opciones de pérdida de clientes disminuye considerablemente el Tiempo Extra de trabajo, pero también disminuye la cantidad de Clientes Atendidos, en la realidad todos los sucesos de perdida son posibles por lo que una buena idea de continuación del proyecto es mejorar el proceso de simulación de dichas perdidas.
- Si los procesos de perdida asumidos, asemejan valores con la realidad, implica que añadirlos disminuye a los Clientes Atendidos a valores cercanos a la mitad de los Clientes Recibidos, pero como ya se dijo estos sucesos de perdida son sucesos reales, por lo que surge una duda interesante: ¿Qué tan cerca están estas estadísticas de la realidad? Normalmente cuando contamos a las personas que están en una cola pensamos solo en los que se quedan luego de entrar a ella, digamos que es una cantidad C, pero en el suceso de Perdida por Cola se tiene un dato interesante de que el mayor porcentaje de perdida esta dado en el Servidor 0 en todas las simulaciones, o sea las personas que se van solo de ver la situación

inicial de la cola, digamos que esta cantidad es PS_0, si decimos que los Clientes Atendidos son una cantidad CA, la idea normal de proporción entre Clientes Atendidos y Recibidos es CA/C, pero la forma en que estamos planteando este problema da una proporción de $CA/(C + PS_0)$, este segundo valor es el que nos interesa encontrar y comparar con nuestros resultados, ¿Qué pasa con esos posibles clientes que no llegan a pasar del inicio?

- Plateando la duda anterior desde otra perspectiva, ¿Si quitáramos PS_0 de la cantidad de Clientes Recibidos, nuestra proporción se asemejaría a la realidad?, lo curioso con las dudas es que si existiera semejanzas en ambas, surge una nueva ¿Se puede demostrar que la variable PS_0 está representada por un porcentaje medible desde el Tiempo de Servicio y la cantidad de Servidores?
- Los resultados más elevados con respecto a Clientes Atendidos son en los que el caso de Perdida por Cola no se consideraba.

Tiempo por cliente					
Exponencial	Normal	Gumbel	Laplace	Logistic	Rayleigh
5.7 // 4.4	2.2 // 1.3	5.5 //4.5	5.8 // 4.9	16.9 // 10.2	11.2 // 8.2
4.2 // 3.6	2.0 // 1.5	4.2 // 3.5	3.9 // 3.3	7.3 // 5.4	5.9 // 4.8
5.1 // 4.8	2.1 // 1.5	5.1 // 5.0	5.0 // 4.9	13.6 // 10.8	9.2 // 8.5
3.1 // 1.1	1.7 // 0.5	3.1 // 1.1	3.0 // 1.1	6.0 // 1.9	10.5 // 16.6
3.6 // 3.3	1.9 // 1.4	3.7 // 3.2	3.6 // 3.1	6.7 // 5.2	5.7 // 5.1
2.5 // 0.9	1.5 // 0.5	2.6 // 0.9	2.5 // 0.9	4.6 // 1.6	3.5 // 1.2
2.6 // 0.9	1.5 // 0.4	2.6 // 0.9	2.5 // 0.9	4.9 // 1.6	3.7 // 1.2
2.3 // 0.9	1.4 // 0.4	2.3 // 0.9	2.3 // 0.8	4.1 // 1.4	3.2 // 1.1

Perdida de clientes					
Exponencial	Normal	Gumbel	Laplace	Logistic	Rayleigh
X	X	X	X	X	X
7.2 // 5.2	3.1 // 2.6	7.5 // 5.3	7.1 // 4.8	16.0 // 9.3	11.8 // 7.7
4.2 // 4.8	3.8 // 4.7	4.1 // 4.8	3.9 // 4.5	4.0 // 4.6	4.1 // 4.9
11.1 // 15.8	12.2 // 14.9	9.7 // 14.7	11.7 // 15.6	10.0 // 17.5	10.5 // 16.6
6.8 // 4.3	4.1 // 3.0	7.0 // 4.1	6.8 // 4.1	13.8 // 8.0	10.6 // 6.3
10.4 // 11.3	8.8 // 9.8	10.6 // 11.7	14.5 // 15.4	4.6 // 1.6	10.9 // 12.4
11.3 // 10.4	11.2 // 8.4	11.9 // 10.7	11.1 // 10.6	11.6 // 13.7	11.4 // 11.8
10.0 // 10.0	8.8 // 8.5	9.6 // 9.5	10.3 // 9.9	11.9 // 12.7	11.8 // 12.0

Conclusiones:

- Las distribuciones logistic y rayleigh son menos eficientes con respecto a tiempo, mientras que la normal es la más eficiente, así que es mejor tener servidores normales ☺
- El máximo para todas las variables aleatorias con respecto a tiempo se encuentra en el caso en que no hay pérdida de clientes y se trabaja un tiempo mayor ¿Será posible que las variables aleatorias también se cansen?
- No encontramos patrones esperados en la pérdida de clientes que lo relacionaran con el tiempo de las variables aleatorias, también pensábamos encontrar poca variación entre las pérdidas de las distribuciones en los casos con menos pérdida de clientes, se cumple para el caso 3 (caso de menor pérdida de clientes) y para el caso 7 (caso de mayor pérdida de clientes), así que no encontramos como analizarlo.

Ahora veamos qué ocurre en los casos en los que se aumenta la cantidad de servidores o el tiempo de simulación, todos los datos están en una tabla de excel que se adjuntó en el repositorio.

Conclusiones para el aumento de servidores:

- El Tiempo Extra aumento para todos los casos con respecto a 5 servidores, tanto para 10 como para 15 servidores, pero la comparación de este parámetro entre 10 y 15 servidores muestra inconsistencia para definir qué caso es menos eficiente.
- Con respecto a Clientes Atendidos hay que resaltar los casos 4 y 7 porque en el primero las cifras para 10 y 15 servidores son mayores que para 5, lo que es posible

dado que es el caso donde solo está habilitada la Perdida por Cola, al haber más servidores los clientes que están en el establecimiento se encuentras más dispersos, para el caso de 7 la Perdida por Retroceso es similar para 10 y 15 servidores pero la Perdida por Cola es menor en el caso de 15, suponemos que por las mismas razones que el caso ya mencionado.

- En la conclusión anterior se resaltaron estos casos porque en el resto se cumple $CA_5 > CA_{10} > CA_{15}$.
- Con respecto a las pérdidas se cumple en los casos que las presentan:

$PE_5 < PE_{10} < PE_{15}$

$PR_5 < PR_{10} < PR_{15}$ (excepto en el caso 7 donde se cumple $PR_{10} > PR_{15}$)

$PC_5 > PC_{15}$ y $PC_5 \geq PC_{10}$ (excepto en el caso 7 donde se cumple $PC_5 < PC_{10}$)

- De esto podemos obtener datos interesantes con respecto al aumento de servidores, aumentarlos hace que aumente el Tiempo Extra, que disminuyan los Clientes Atendidos y tenemos que los principales motivos es porque se pierden más clientes en Espera y Retroceso, dado que ahora hay más servidores por los que pasar, cada cliente se ve en más situaciones donde puede sufrir retroceso y dado que tiene más servidores por recorrer debe demorarse más tiempo en el establecimiento, si bien están más dispersos y provoca una disminución en la Perdida por Cola, no es suficiente para solventar las dos perdidas ya mencionadas. A la hora de crear un sistema es un aspecto que debe considerarse para obtener el valor óptimo para dicho sistema.
- Comparaciones interesantes con respecto a la pérdida de clientes por distribución:

Caso 2: Normal < Laplace < Exponencial ≤ Gumbel < Rayleigh < Logistic, se cumple para las 3 cantidades de servidores

Caso 3 y 7: Presentan poca variabilidad entre los valores de cada respectivo experimento

Caso 5: Normal < Laplace ≤ Exponencial < Gumbel < Rayleigh < Logistic, se cumple para las 3 cantidades de servidores
Caso 8: Normal < Gumbel ≤ Exponencial < Laplace < Rayleigh < Logistic, se cumple para 5 y 10 servidores
- Con respecto al tiempo promedio de clientes por distribución se hace notar que el Caso 1 sigue siendo el de mayores valores para todas las variables en sus respectivos experimentos.

Conclusiones para el aumento de tiempo:
- Se encontró un patrón en la Perdida por Espera: $C_8 < C_6 < C_5 < C_2$ para las 3 simulaciones.
- Con respecto a la pérdida de clientes por distribución se puede ver que en el Caso 3 vuelve a existir poca variación
- Se mantiene que el tiempo mayor de espera del cliente promedio con respecto a la distribución se encuentra en el Caso 1.