PROYECTO FINAL

"VALLET PARKING"

MODELOS COMPUTACIONALES

INTEGRANTES:

CAMACHO CANIZALES JUAN ABIMAEL

ESTRADA CASTILLO ERIKA RUBY

DOCENTE:

VALERIA SOTO MENDOZA

24/11/2022

DESCRIPCION GENERAL DEL PROYECTO

El proyecto en general se basa en un trabajo realizado por un vallet parking el cual se realiza en una seria de lugares como hoteles, restaurantes, eventos privados, etc, en pocas palabras El valet parking es una variante del estacionamiento convencional en el que tu comprador deja su vehículo en la entrada de tu empresa para que un empleado lo lleve al aparcamiento, a este trabajador se le denomina Valet parking y también se encarga de devolvérselo al cliente una vez que decida retirarse de tu negocio.

Como puntos principales que nos ayuda un Vallet Parking:

- Mover los otros vehículos cuando no hay espacio para salir o entrar.
- Mantener en buen estado los vehículos de tu cliente.
- Cuidar de las llaves del vehículo estacionado.
- Escoger la mejor plaza de parqueo para el vehículo.

Un Valet Parking sirve para que tus clientes tengan mayor comodidad a la hora de estacionar sus vehículos en tu empresa, pues en lugar de pasar varios minutos buscando un sitio en donde parquear alguien se encargará de encontrar el lugar más apropiado. También sirve para darle más seguridad a tu parqueadero, pues los únicos con acceso a este serán empleados de tu confianza.

En el proyecto se basa específicamente en buscar un mayor control entre clientes y vallet´s poder conocer cuantas personas se necesitan para poder cubrir un evento.

Se entiende por Teoría de Colas el estudio de las líneas de espera que se producen cuando llegan clientes demandando un servicio, esperando si no se les puede atender inmediatamente y partiendo cuando ya han sido servidos.

Observando el Vallet de esta manera se puede aplicar al modelo viendo los carros como clientes y a los vallet´s como servidores, todo el proceso que se lleva es el servicio

METAS DEL PROYECTO

- ✓ Se quiere llegar a saber cuantos servidores se necesitan para cubrir un evento
- ✓ Conocer los tiempos del servicio
- ✓ Mejorar la calidad del servicio en cuanto a no tener esperando al cliente por mayor tiempo
- ✓ Aprender el funcionamiento de colas para situaciones reales

DESCRIPCION DEL MODELO

La teoría de colas es el estudio de los sistemas de líneas de espera en sus distintas modalidades. El estudio de estos modelos sirve para determinar la forma más efectiva de gestionar un sistema de colas.

La teoría de colas es el estudio matemático del comportamiento de líneas de espera. Esta se presenta cuando los clientes llegan a un lugar demandando un servicio a un «servidor», el cual tiene una cierta capacidad de atención. Si el servidor no está disponible inmediatamente y el cliente decide esperar, entonces se forma la línea de espera.

¿PARA QUÉ SIRVE LA TEORÍA DE COLAS?

El estudio de las colas nos sirve para proporcionar tanto una base teórica del tipo de servicio que podemos esperar de un determinado recurso, como la forma en la cual dicho recurso puede ser diseñado para proporcionar un determinado grado de servicio a sus clientes.

Aplicaciones de la teoría de colas:

- ✓ Facturación en aeropuertos.
- ✓ Cajeros automáticos.
- ✓ Restaurantes de comida rápida.
- ✓ Esperas en líneas de atención telefónica.
- ✓ Intersecciones de tráfico.
- ✓ Aviones en espera para aterrizar.
- ✓ Llamadas a la policía o a compañías de servicios públicos.
- ✓ Estándares de calidad del servicio.
- ✓ Análisis económicos que incluyan comparaciones entre costes de explotación, inversiones de capital.

OBJETIVOS DE LA TEORÍA DE COLAS

Identificar el nivel óptimo de capacidad del sistema que minimiza el coste global del mismo.

Evaluar el impacto que las posibles alternativas de modificación de la capacidad del sistema tendrían en el coste total del mismo.

Establecer un balance equilibrado («óptimo») entre las consideraciones cuantitativas de costes y las cualitativas de servicio.

Hay que prestar atención al tiempo de permanencia en el sistema o en la cola: la «paciencia» de los clientes depende del tipo de servicio específico considerado y eso puede hacer que un cliente «abandone» el sistema.

VENTAJAS Y DESVENTAJAS DE LA TEORÍA DE COLAS

Ventajas

- Los modelos de colas implican siempre aproximaciones a la realidad y una simplificación de ésta.
- Los resultados permiten apreciar el orden de importancia, los cambios con relación a un punto de referencia y las tendencias más probables.
- Resultados «cerrados» limitados casi siempre a situaciones de "estado estacionario» y obtenidos sobre todo (aunque no exclusivamente) para su aplicación a sistemas de nacimiento y muerte y de "fase".
- Proporciona algunas cotas útiles para sistemas más generales en estado estacionario.
- Cada vez hay más soluciones numéricas disponibles para sistemas dinámicos.

Desventajas

- Los modelos de simulación en una computadora son costosos y requieren mucho tiempo para desarrollarse y validarse.
- Se requiere gran cantidad de corridas computacionales para encontrar "soluciones óptimas", lo cual repercute en altos costos.
- Es difícil aceptar los modelos de simulación.
- Los modelos de simulación no dan soluciones óptimas
- La solución de un modelo de simulación puede dar al analista un falso sentido de seguridad.

CodigoFuenteProyecto_CamachoJuan_EstradaErika

Juan Abimael Camacho Canizales

Erika Ruby Estrada Castillo

2022-11-24

```
library(queueing)
```

Primer Ejemplo: Evento pequeño de 150 vehiculos. -Aproximado de Vehiculos por hora: 50 vehiculos -Tiempo en que el corredor deja el vehiculo: 4 minutos -Cantidad de coches que el corredor puede llevar por hora: 15 vehiculos

```
mmckm1a <- NewInput.MMCKM(lambda = 50, mu = 15, c = 1, k = 150, m = 150)
mmc0a <- QueueingModel(mmckm1a)

mmckm1b <- NewInput.MMCKM(lambda = 50, mu = 15, c = 2, k = 150, m = 150)
mmc0b <- QueueingModel(mmckm1b)

mmckm1c <- NewInput.MMCKM(lambda = 50, mu = 15, c = 3, k = 150, m = 150)
mmc0c <- QueueingModel(mmckm1c)

mmckm1d <- NewInput.MMCKM(lambda = 50, mu = 15, c = 4, k = 150, m = 150)
mmc0d <- QueueingModel(mmckm1d)

mmckm1e <- NewInput.MMCKM(lambda = 50, mu = 15, c = 5, k = 150, m = 150)
mmc0e <- QueueingModel(mmckm1e)

mmckm1f <- NewInput.MMCKM(lambda = 50, mu = 15, c = 25, k = 150, m = 150)
mmc0f <- QueueingModel(mmckm1f)

CompareQueueingModels(mmc0a, mmc0b, mmc0c, mmc0d, mmc0e, mmc0f)
```

```
##   lambda mu  c   k   m RO            P0    Lq        Wq   X     L
W
## 1     50 15  1 150 150  1  0.000000e+00 148.7 9.9133333  15 149.7 9.98
0000
## 2     50 15  2 150 150  1 2.536223e-297 147.4 4.9133333  30 149.4 4.98
0000
## 3     50 15  3 150 150  1 2.164739e-271 146.1 3.2466667  45 149.1 3.31
3333
## 4     50 15  4 150 150  1 3.724872e-253 144.8 2.4133333  60 148.8 2.48
0000
## 5     50 15  5 150 150  1 3.887630e-239 143.5 1.9133333  75 148.5 1.98
0000
## 6     50 15 25 150 150  1 3.070669e-145 117.5 0.3133333 375 142.5 0.38
0000
##         Wqq   Lqq
## 1        NA    NA
```

```
## 2 4.9133333 147.4
## 3 3.2466667 146.1
## 4 2.4133333 144.8
## 5 1.9133333 143.5
## 6 0.3133333 117.5

#CLIENTES ESPERADOS
Ls <- c(L(mmc0a),L(mmc0b),L(mmc0c),L(mmc0d), L(mmc0e), L(mmc0f))
Ls

## [1] 149.7 149.4 149.1 148.8 148.5 142.5

#Productividad
Productividad <- ((150 - Ls)/150)*100
Productividad

## [1] 0.2 0.4 0.6 0.8 1.0 5.0

#Tiempo promedio en el sistema
tiempo <- c(W(mmc0a),W(mmc0b),W(mmc0c),W(mmc0d), W(mmc0e), W(mmc0f))*60 #
Minutos
tiempo

## [1] 598.8 298.8 198.8 148.8 118.8  22.8

#Tiempo promedio en cola
tiempoCola <- c(Wq(mmc0a),Wq(mmc0b),Wq(mmc0c),Wq(mmc0d),Wq(mmc0e),Wq(mmc0
f))*60 #Minutos
tiempoCola

## [1] 594.8 294.8 194.8 144.8 114.8  18.8
```

Segundo Ejemplo: Evento mediano de 300 vehiculos. -Aproximado de Vehiculos por hora: 70 vehiculos -Tiempo en que el corredor deja el vehiculo: 4 minutos -Cantidad de coches que el corredor puede llevar por hora: 15 vehiculos

```
mmckm3a <- NewInput.MMCKM(lambda = 70, mu = 15, c = 1, k = 300, m = 300)
mmc1a <- QueueingModel(mmckm3a)

mmckm3b <- NewInput.MMCKM(lambda = 70, mu = 15, c = 2, k = 300, m = 300)
mmc1b <- QueueingModel(mmckm3b)

mmckm3c <- NewInput.MMCKM(lambda = 70, mu = 15, c = 3, k = 300, m = 300)
mmc1c <- QueueingModel(mmckm3c)

mmckm3d <- NewInput.MMCKM(lambda = 70, mu = 15, c = 4, k = 300, m = 300)
mmc1d <- QueueingModel(mmckm3d)

mmckm3e <- NewInput.MMCKM(lambda = 70, mu = 15, c = 5, k = 300, m = 300)
mmc1e <- QueueingModel(mmckm3e)

mmckm3f <- NewInput.MMCKM(lambda = 70, mu = 15, c = 25, k = 300, m = 300)
```

```
mmc1f <- QueueingModel(mmckm3f)

CompareQueueingModels(mmc1a, mmc1b, mmc1c, mmc1d, mmc1e, mmc1f)

##   lambda mu  c   k   m RO  P0       Lq         Wq   X        L         
W Wqq
## 1     70 15  1 300 300  1   0 298.7857 19.9190476  15 299.7857 19.9857
143  NA
## 2     70 15  2 300 300  1   0 297.5714  9.9190476  30 299.5714  9.9857
143  NA
## 3     70 15  3 300 300  1   0 296.3571  6.5857143  45 299.3571  6.6523
810  NA
## 4     70 15  4 300 300  1   0 295.1429  4.9190476  60 299.1429  4.9857
143  NA
## 5     70 15  5 300 300  1   0 293.9286  3.9190476  75 298.9286  3.9857
143  NA
## 6     70 15 25 300 300  1 NaN 269.6429  0.7190476 375 294.6429  0.7857
143  NA
##   Lqq
## 1  NA
## 2  NA
## 3  NA
## 4  NA
## 5  NA
## 6  NA

Ls1 <- c(L(mmc1a),L(mmc1b),L(mmc1c),L(mmc1d), L(mmc1e), L(mmc1f))
Ls1

## [1] 299.7857 299.5714 299.3571 299.1429 298.9286 294.6429

#Productividad
Productividad1 <- ((300 - Ls1)/300)*100
Productividad1

## [1] 0.07142857 0.14285714 0.21428571 0.28571429 0.35714286 1.78571429

#Tiempo promedio en el sistema
tiempo2 <- c(W(mmc1a),W(mmc1b),W(mmc1c),W(mmc1d),W(mmc1e),W(mmc1f))*60 #M
inutos
tiempo2

## [1] 1199.14286  599.14286  399.14286  299.14286  239.14286   47.14286

#Tiempo promedio en cola
tiempoCola2 <- c(Wq(mmc1a),Wq(mmc1b),Wq(mmc1c),Wq(mmc1d),Wq(mmc1e),Wq(mmc
1f))*60 #Minutos
tiempoCola2

## [1] 1195.14286  595.14286  395.14286  295.14286  235.14286   43.14286
```

Tercer Ejemplo: Evento grande de 500 vehiculos. -Aproximado de Vehiculos por hora: 150 vehiculos -Tiempo en que el corredor deja el vehiculo: 4 minutos -Cantidad de coches que el corredor puede llevar por hora: 15 vehiculos

```
mmckm5a <- NewInput.MMCKM(lambda = 150, mu = 15, c = 1, k = 500, m = 500)
mmc2a <- QueueingModel(mmckm5a)

mmckm5b <- NewInput.MMCKM(lambda = 150, mu = 15, c = 2, k = 500, m = 500)
mmc2b <- QueueingModel(mmckm5b)

mmckm5c <- NewInput.MMCKM(lambda = 150, mu = 15, c = 3, k = 500, m = 500)
mmc2c <- QueueingModel(mmckm5c)

mmckm5d <- NewInput.MMCKM(lambda = 150, mu = 15, c = 4, k = 500, m = 500)
mmc2d <- QueueingModel(mmckm5d)

mmckm5e <- NewInput.MMCKM(lambda = 150, mu = 15, c = 5, k = 500, m = 500)
mmc2e <- QueueingModel(mmckm5e)

mmckm5f <- NewInput.MMCKM(lambda = 150, mu = 15, c = 25, k = 500, m = 500
)
mmc2f <- QueueingModel(mmckm5f)

CompareQueueingModels(mmc2a, mmc2b, mmc2c, mmc2d, mmc2e, mmc2f)

##   lambda mu  c   k   m RO  P0    Lq        Wq   X     L         W Wqq
Lqq
## 1    150 15  1 500 500  1   0 498.9 33.260000  15 499.9 33.326667  NA
NA
## 2    150 15  2 500 500  1   0 497.8 16.593333  30 499.8 16.660000  NA
NA
## 3    150 15  3 500 500  1   0 496.7 11.037778  45 499.7 11.104444  NA
NA
## 4    150 15  4 500 500  1   0 495.6  8.260000  60 499.6  8.326667  NA
NA
## 5    150 15  5 500 500  1 NaN 494.5  6.593333  75 499.5  6.660000  NA
NA
## 6    150 15 25 500 500  1 NaN 472.5  1.260000 375 497.5  1.326667  NA
NA

Ls3 <- c(L(mmc2a),L(mmc2b),L(mmc2c),L(mmc2d), L(mmc2e), L(mmc2f))
Ls3

## [1] 499.9 499.8 499.7 499.6 499.5 497.5

#Productividad
Productividad3 <- ((500 - Ls3)/500) * 100
Productividad3

## [1] 0.02 0.04 0.06 0.08 0.10 0.50
```

```r
#Tiempo promedio en el sistema
tiempo3 <- c(W(mmc2a),W(mmc2b),W(mmc2c),W(mmc2d),W(mmc2e),W(mmc2f))*60 #M
inutos
tiempo3

## [1] 1999.6000  999.6000  666.2667  499.6000  399.6000   79.6000

#Tiempo promedio en cola
tiempoCola3 <- c(Wq(mmc2a),Wq(mmc2b),Wq(mmc2c),Wq(mmc2d),Wq(mmc2e),Wq(mmc
2f))*60 #Minutos
tiempoCola3

## [1] 1995.6000  995.6000  662.2667  495.6000  395.6000   75.6000
```

CONCLUSIONES

Las conclusiones del proyecto se presentan en dos situaciones, gracias a la observación en las medidas que utilizamos y en como está repartida la información en nuestra problemática, podemos ver que lo que se conoce como productividad es algo que a números tan altos se puede entender poco o se puede ver que no hay mucha diferencia entre una cantidad de servidores a otro, esto se ve mucho cuando el aumento de servidores es de 1 en 1. En cuanto a la métrica de tiempos, entendemos que hay situaciones que pueden alterar los tiempos y en que en cuestiones tan complejas como los movimientos de vehículo que pueden ser mas complicados tal vez por experiencia, situaciones ambientales o laborales, el sistema puede fallar en métricas, pero los datos son un aproximado de un sistema ideal donde no haya estas complejidades.