Universidad de La Habana

Facultad de Matemática y
Computación

Proyecto de Simulación basado en Eventos Discretos

Estudiante:
Laura Brito Guerrero C-412

0.1. Orden del problema asignado: La cocina de Kojo (Kojo's Kitchen)

La cocina de Kojo es uno de los puestos de comida rápida en un centro comercial. El centro comercial está abierto entre las 10:00 am y las 9:00 pm cada día. En este lugar se sirven dos tipos de productos: sándwiches y suchi. Para los objetivos de este proyecto se asumirá que existen solo dos tipos de consumidores: unos consumen solo sándwiches y los otros consumen solo productos de la gama del suchi. En Kojo hay dos períodos de hora pico durante un día de trabajo; uno entre las 11:30 am y la 1:30 pm, y el otro entre las 5:00 pm y las 7:00 pm. El intervalo de tiempo entre el arribo de un consumidor y el de otro no es homogéneo pero, por conveniencia, se asumirá que es homogéneo. El intervalo de tiempo de los segmentos homogéneos, distribuye de forma exponencial.
Actualmente dos empleados trabajan todo el día preparando sándwiches y suchi para los consumidores. El tiempo de preparación depende del producto en cuestión. Estos distribuyen de forma uniforme, en un rango de 3 a 5 minutos para la preparación de sándwiches y entre 5 a 8 minutos para la preparación del suchi.
El administrador de Kojo está muy feliz con el negocio, pero ha estado recibiendo quejas de los consumidores por la demora de sus peticiones. Él está interesado en explorar algunas opciones de distribución del personal para reducir el número de quejas. Su interés está centrado en comparar la situación actual con una opción alternativa donde se emplea un tercer empleado durante los períodos más ocupados. La medida del desempeño de estas opciones estará dada por el porciento de consumidores que espera más de 5 minutos por un servicio durante el curso de un día de trabajo.
Se desea obtener el porciento de consumidores que esperan más de 5 minutos cuando solo dos empleados están trabajando y este mismo dato agregando un empleado en las horas picos.

0.2. Principales ideas seguidas para la solución del problema

Primeramente para la solución de este problema se deben tener en cuenta que se quiere llegar a dos resultados, por lo tanto se deben dar respuestas a dos problemáticas y comparar sus resultados. La llegada de los clientes distribuye exponencial, como lo dice la orden, por lo tanto, se simulan en minutos la apertura de los mismos. Con respecto al tiempo, se convierte el horario en el cual se encuentra abierta la cocina en minutos para realizar el análisis más fácil de concebir, luego se convierte en la hora exacta donde ocurren las entradas y salidas de los clientes. Para cada solución nos auxiliamos de un diccionario (**time_mean**) para guardar a los clientes que demoran más de 5 minutos esperando su pedido, este dato se conoce ya que cada cliente tiene guardado su tiempo de llegada (**A**) y su tiempo de salida (**D**), por ejemplo, el cliente i que

pide sándwich (suchi) pertenece a **time_mean** si y solo si

$$D[i] - A[i] - 5 \geq 5(D[i] - A[i] - 8 \geq 5). \tag{1}$$

.
Una vez que la simulación termine, en **time_mean** estará el total de clientes que espera más de 5 minutos, por lo tanto, con este dato y con el total de clientes que asiste a la cocina (**Na**) se calcula el porciento que representa. Se denota a la cantidad de clientes que están en **time_mean** como **x**, por lo tanto el porciento se calcula mediante la expresión:

$$\frac{100x}{Na}. \tag{2}$$

Con respecto a las horas picos, se define para la hora de llegada del próximo cliente (**ta**) otra distribución exponencial con

$$\lambda = \frac{1}{5}, \tag{3}$$

y las restantes horas se definen con la distribución exponencial siendo

$$\lambda = \frac{1}{60}. \tag{4}$$

A continuación se muestra las ideas a seguir para cada problema en cuestión.

1. **Cuando en la cocina solo atienden dos empleados:**

   a) Como cada empleado se dedica a un tipo de elaboración específica: sándwich o suchi, se analizan por separado, ya que la actividad de uno no interfiere en el otro, por ello se tendrán dos colecciones de datos (**Na_1** para el sándwich y **Na_2** para el suchi) para guardar a los clientes que les interesa cada elaboración y están en la cola esperando por su pedido.

   b) Se guardan en dos variables (**t1** para el sándwich y **t2** para el suchi) el tiempo de la partida de los clientes, cada momento siendo consecuente con el tiempo que termina el empleado de cada elaboración de terminar un pedido. **t1** distribuye uniforme de 3 a 5 minutos y **t2** también distribuye uniforme de 5 a 8 minutos.

2. **Cuando se emplea un tercer empleado en los horarios picos**

   a) El razonamiento se mantiene análogo al anterior problema con la distinción de que en los horarios picos varía su razonamiento. Se sigue el siguiente planteamiento:

      1) En los horarios picos, al llegar un cliente que elige la elaboración i (i = sándwich, suchi), si el empleado encargado de la elaboración i está ocupado entonces el nuevo empleado se encarga del pedido si él está desocupado,

2) de lo contrario el cliente espera en la misma cola que todos los demás clientes y espera a ser atendido cuando le toque el turno,
3) cuando le toque el turno analizar si el empleado encargado de la elaboración i está desocupado, de ser cierto es atendido por el mismo, si no, analiza si el nuevo empleado está disponible, si se cumple es atendido por él, de lo contrario sigue pendiente el cliente en la cola hasta que se libere el primero de los empleados mencionados.

b) Se guardan en tres variables (**t1** para el sándwich, **t2** para el suchi y **t3** para el nuevo empleado que realiza tanto sándwich como suchi) el tiempo de la partida de los clientes, cada momento siendo consecuente con el tiempo que termina el empleado de cada elaboración de terminar un pedido. **t1** distribuye uniforme de 3 a 5 minutos, **t2** también distribuye uniforme de 5 a 8 minutos y **t3** distribuye depende del pedido que esté elaborando.

En el próximo epígrafe se explica en base del pseudocódigo implementado la simulación utilizada en cada caso.

0.3. Modelo de Simulación de Eventos Discretos desarrollado para resolver el problema

Cada problemática se resuelve mediante la implementación de simulación basada en eventos discretos. Se describe cada uno por separado y se explica el sistema utilizado en cuestión.

0.3.1. Cuando en la cocina solo atienden dos empleados

Se implementa un sistema de atención de dos servidores. Existen dos servidores (el servidor del sándwich y el del suchi), y una vez que llega un cliente es atendido por el servidor correspondiente, si está vacío sino espero su turno en la cola correspondiente. Cuando alguno de los servidores termina de atender a un cliente comienza inmediatamente a atender el cliente que lleva más tiempo esperando en la cola de su servicio.
Existe un tiempo fijo $\mathbf{T} = 660$ minutos, que equivale a las 11 horas que se encuentra abierto el puesto de comida, a partir del cual no se le permite entrar a más ningún cliente al sistema, sin embargo todos los clientes que ya estaban dentro deberán ser atendidos.

Las variables del sistema son:

1. t: variable del tiempo transcurrido,
2. Na: la cantidad de arribos en el sistema,
3. Nd: la cantidad de partidas en el sistema,

4. Na_1: clientes en la cola para el servicio del sándwich,
5. Na_2: clientes en la cola para el servicio del suchi,
6. n: cantidad de clientes en el sistema
7. A: A[i] es el tiempo de llegada del cliente i,
8. D: D[i] es el tiempo de partida del cliente i,
9. S: S[i] es el servicio que escoge el cliente i (Si S[i] = 1 entonces el cliente i quiere sándwich, si S[i] = 2 el cliente i quiere suchi),
10. Tp: tiempo que atiende el sistema luego de cerrar,
11. ta: tiempo del próximo arribo de un cliente,
12. t1: tiempo de la próxima partida del servicio del sándwich,
13. t2: tiempo de la próxima partida del servicio del suchi.

El **pseudocódigo** de la lógica implementada es el siguiente:

Inicialización

```
t = Na = Nd = n = Tp = 0
Na_1 = Na_2 = []
A = D = S = {}
t1 = t2 = ∞
ta = generar variable con distribución exponencial
```

Caso 1: arriba un nuevo cliente

```
if ta = min(ta, t1, t2) y ta ≤ T - 1 then
    t = ta
    n++, Na++
    if es hora pico then
        Tt = generar variable con distribución exponencial con lambda=1/5
    else
        Tt = generar variable con distribución exponencial con lambda=1/60
    end if
    ta = t + Tt
    A[Na] = t
    select = choice([1, 2])
    S[Na] = select
    if select = 1 then
        if Na_1 no tiene elementos then
            Y1 = generar variable uniforme de 3 a 5 minutos
            t1 = t + Y1
        end if
```

```
        agrega el cliente Na a Na_1
    else
        if Na_2 no tiene elementos then
            Y2 = generar variable uniforme de 5 a 8 minutos
            t2 = t + Y2
        end if
        agrega el cliente Na a Na_2
    end if
end if
```

Caso 2: se va un cliente con su pedido de sandwich(i = 1) o suchi(i = 2)

```
if  ti = min(ta, t1, t2) y ti ≤ T - 1, para i = 1,2 then
    if n ≥ 1 y Na_i no esta vacio then
        t = ti
        n–, Nd++
        N = Na_i[0]
        eliminar N de Na_i
        D[N] = t
        if D[N] - A[N] - 5 ≥ 5 si i = 1, D[N] - A[N] - 8 ≥ 5 si i = 2 then
            time_mean[N] = D[N] - A[N] - 5 si i = 1, D[N] - A[N] - 8 si i = 2
        end if
        if Na_i esta vacio then
            ti = ∞
        else
            Yi = generar variable uniforme correspondiente
            ti = t + Yi
        end if
    else
        ti = ∞
    end if
end if
```

Caso 3: Restaurante cerrado y se va un cliente con su pedido de sandwich(i = 1) o suchi(i = 2)

```
if  ti = min(ta, t1, t2) ≥ T y Na_i no esta vacio, para i = 1,2 then
    t = ti
    n–, Nd++
    N = Na_i[0]
    eliminar N de Na_i
    D[N] = t
    if D[N] - A[N] - 5 ≥ 5 si i = 1, D[N] - A[N] - 8 ≥ 5 si i = 2 then
        time_mean[N] = D[N] - A[N] - 5 si i = 1, D[N] - A[N] - 8 si i = 2
    end if
    if Na_i esta vacio then
        ti = ∞
    else
```

Yi = generar variable uniforme correspondiente
ti = t + Yi
end if
end if

Caso 4: Restaurante cerrado y ya no quedan clientes por atender

if min(ta, t1, t2) ≥ T y n = 0 **then**
Tp = max(t - T, 0)
BREAK
end if

La solución se localiza en la dirección del proyecto *code/src/kojo_kitchen_1.py*. Para poder observar la simulación descomente los **print()** del código.

0.3.2. Cuando se emplea un tercer empleado en los horarios picos

Se implementa un sistema de atención con dos servidores en paralelo, donde el primer servidor funciona como un servidor de dos servidores independientes. Cuando un cliente llega este es atendido por el servidor que se encuentre vacío y si los dos están llenos entonces espera en la cola. Se denomina que el primer servidor está vacío para el cliente i que pide la elaboración j si el subservidor que elabora el pedido j está vacío. Se dice que el segundo servidor está vacío cuando el tiempo de arribo del cliente i se encuentra comprendido en el horario pico.
Las variables del sistema son:

1. t: variable del tiempo transcurrido,
2. SS: variable de estado del sistema,
3. Na: la cantidad de arribos en el sistema,
4. Ci: la cantidad de partidas en el sistema por el servidor i, i = 1, 2, 3 (para i = 1, 2 se refiere al mismo servidor lo que se diferencian en el mismo los servicios prestados, i = 1 para el sándwich e i = 2 para el suchi),
5. A: A[i] es el tiempo de llegada del cliente i,
6. D: D[i] es el tiempo de partida del cliente i,
7. S: S[i] es el servicio que escoge el cliente i (Si S[i] = 1 entonces el cliente i quiere sándwich, si S[i] = 2 el cliente i quiere suchi),
8. ta: tiempo del próximo arribo de un cliente,
9. t1: tiempo de la próxima partida del servicio del sándwich para el servidor 1,
10. t2: tiempo de la próxima partida del servicio del suchi para el servidor 1,

11. t3: tiempo de la próxima partida del servicio del sándwich o suchi para el servidor 2.

Para mayor visualización se muestra el pseudocódigo. **Inicialización**

t = Na = C1 = C2 = C3 = 0
SS = [0]
A = D = S = {}
t1 = t2 = t3 = ∞
ta = generar variable con distribución exponencial

Caso 1: arriba un nuevo cliente

if ta = min(ta, t1, t2, t3) y ta $\leq$ T - 1 **then**
    t = ta
    Na++
    if es hora pico **then**
        Tt = generar variable con distribución exponencial con lambda=1/5
    else
        Tt = generar variable con distribución exponencial con lambda=1/60
    end if
    ta = t + Tt
    A[Na] = t
    select = choice([1, 2])
    S[Na] = select
    if SS = [0] **then**
        if select = 1 **then**
            SS = [1, Na, 0, 0]
            Y1 = generar variable uniforme de 3 a 5 minutos
            t1 = t + Y1
        else
            SS = [1, 0, Na, 0]
            Y2 = generar variable uniforme de 5 a 8 minutos
            t2 = t + Y2
        end if
    else if select = 1 y SS = [n, 0, i2, i3] **then**
        SS = [n + 1, Na, i2, i3]
        Y1 = generar variable uniforme de 3 a 5 minutos
        t1 = t + Y1
    else if select = 2 y SS = [n, i1, 0, i3] **then**
        SS = [n + 1, i1, Na, i3]
        Y2 = generar variable uniforme de 5 a 8 minutos
        t2 = t + Y2
    else if es hora pico y SS = [n, i1, i2, 0] **then**
        SS = [n + 1, i1, i2, i3]
        Yi = generar variable uniforme correspondiente
        t3 = t + Yi
    else if SS = [n, i1, i2, i3, i4, . . . , in] **then**

SS = [n + 1, i1, i2, i3, i4, . . ., in, Na]
end if
end if

Caso 2: se va un cliente con su pedido de sandwich(i = 1) o suchi(i = 2) del servidor 1

if ti = min(ta, t1, t2, t3) y ti ≤ T - 1, para i = 1,2 **then**
t = ti
Ci++
N = SS[i]
D[N] = t
if D[N] - A[N] - 5 ≥ 5 si i = 1, D[N] - A[N] - 8 ≥ 5 si i = 2 **then**
time_mean[N] = D[N] - A[N] - 5 si i = 1, D[N] - A[N] - 8 si i = 2
end if
if SS = [1, i1, 0, 0] si i = 1, SS = [1, 0, i2, 0] si i = 2 **then**
SS = [0]
ti = ∞
else if SS = [n, i1, i2, i3] **then**
SS = [n - 1, 0, i2, i3] si i = 1, SS = [n - 1, i1, 0, i3] si i = 2
ti = ∞
else
ik = cliente que lleva mas tiempo en la cola esperando por pedir sandwich(i = 1), o suchi(i = 2)
SS = [n - 1, ik, i2, i3, i4, . . . , i(n-1)] para i = 1, SS = [n - 1, i1, ik, i3, i4, . . . , i(n-1)] para i = 2
Yi = generar variable uniforme correspondiente para servicio i
ti = t + Yi
end if
end if

Caso 3: se va un cliente con su pedido de sandwich o suchi del servidor 2

if t3 = min(ta, t1, t2, t3) y t3 ≤ T - 1 **then**
t = t3
C3++
N = SS[3]
D[N] = t
if D[N] - A[N] - 5 ≥ 5 si S[N] = 1, D[N] - A[N] - 8 ≥ 5 si S[N] = 2 **then**
time_mean[N] = D[N] - A[N] - 5 si S[N] = 1, D[N] - A[N] - 8 si S[N] = 2
end if
if SS = [1, 0, 0, i3] **then**
SS = [0]
t3 = ∞
else if SS = [n, i1, i2, i3] **then**
SS = [n - 1, i1, i2, 0]
t3 = ∞

```
    else
        SS = [n - 1, i1, i2, i4, i5, . . . , i(n-1)]
        Yi = generar variable uniforme correspondiente de S[i4]
        t3 = t + Yi
    end if
end if
```

Caso 4: Restaurante cerrado y ya no quedan clientes por atender

```
if min(ta, t1, t2, t3) ≥ T y SS = [0] then
    BREAK
end if
```

La solución se localiza en la dirección del proyecto *code/src/kojo_kitchen_opt.py*. Para poder observar la simulación descomente los **print()** del código.

0.4. Consideraciones obtenidas a partir de la ejecución de las simulaciones del problema

A partir de la simulación de estas situaciones se evidencia que el promedio de los clientes que esperan más de 5 minutos por su pedido es superior cuando solo hay dos empleados trabajando, sin embargo cuando existe otro empleado esta espera es prácticamente nula en la mayoría de los casos.
La implementación de la generación de las variables aleatorias se encuentran en *code/src/utils/random_var_generation.py*.
La ejecución de todo el programa y la comparación de los dos métodos está en *code/src/main.py*.