Universidad Simón Bolívar
Departamento de Computación y
Tecnología de la Información
CI-3825: Sistemas de Operación I

Prof. Fernando Torre Mora

Proyecto 1 Enero – Marzo 2023

Transporte USB

1 Introducción

Uno de los principales obstáculos para volver a la presencialidad es la insuficiencia del servicio de transporte universitario. Sin embargo, no está establecido cuánto es suficiente. Para encontrar este nivel de suficiencia, se le pide realizar un programa que simule un día de servicio que indique el nivel de suficiencia de diferentes configuraciones, usando procesos para las rutas e hilos para los autobuses.

La suficiencia o insuficiencia del servicio se medirá en el número de usuarios que logra o no logra llegar a tiempo. En esta simulación, supondremos que los usuarios llegan a la parada exactamente 1½ horas antes de tener que estar en la universidad (independientemente de la cola esperada o la longitud de la ruta) solamente una vez por hora y en grupo. Por lo tanto, la insuficiencia del sistema es el número de usuarios que tardaron más de 1½ horas desde que llegaron a la parada hasta que llegaron a la universidad.

El tamaño del grupo que corresponde a cada parada en cada hora, viene indicado en un archivo que denominaremos “caracterización de la carga al sistema”. Las horas en la que los autobuses parten a cada parada vienen indicadas en un archivo que denominaremos “caracterización del servicio”.

Para simplificar, adicionalmente supondremos que:

- Los autobuses transportan a los usuarios en un solo sentido: desde la parada hacia la universidad (si el servicio es suficiente en un sentido, supondremos que es suficiente en el sentido contrario)
- Los autobuses siempre parten desde la universidad (no hay depósitos alternos)
- Al llegar un autobús, se montan tantos usuarios como el autobús tenga capacidad (Si hay menos usuarios en la parada que la capacidad del autobús, se montan todos. Si hay más, los que no “caben” no se montan)
- No hay distinción entre tipos de usuario (no hay trato prioritario para embarazadas, etc.)
- Si un usuario no se logra montar, esperará indefinidamente (no “se va” luego de un tiempo)
- El autobús siempre pasa 10 minutos en la parada (independientemente si monta 5 usuarios ó 50)
- La velocidad del autobús es constante en su recorrido
- Cada autobús es usado una sola vez en el día en el sentido parada>USB
- El servicio sólo opera 8 horas
- Si hay usuarios en la parada al terminar el horario de operación del servicio, ninguno de ellos llega a tiempo (sin importar en qué momento llegaron a la parada)
- Si una parada tiene carga, pero no servicio, esos usuarios no usan el transporte (no intentan montarse en otra parada)

El programa debe imprimir el progreso de los autobuses mientras corre la simulación. Al terminar, debe imprimir cuántos usuarios llegaron a tiempo en cada ruta, y cuántos tardaron más de 1½ horas

2 Requerimientos del programa

Debe crear un programa que pueda ejecutarse desde la cónsola con el comando

```
$ ./simutransusb -s <archivo> [-c <archivo>] [-t <num>]
```

donde

- `-s` indica el nombre del archivo de caracterización del servicio (si no se especifica ninguno, debe darse un error)
- `-c` indica el nombre del archivo de caracterización de la carga (si no se especifica ninguno, se debe suponer el archivo `carga.csv`)
- `-t` indica el tiempo en segundos que debe durar un minuto en la simulación (si no se especifica ninguno, se debe suponer 0,25 segundos)

2.1 Formato de salida

El programa debe imprimir el estado de la simulación cada minuto simulado en el siguiente formato:

```
HH:MM
P_1: u_1 "[" (">"|"<")^{t1,1} (" ")^{10-t1,1} "]"_{B1,1} ... "[" (">"|"<")^{tn1,1} (" ")^{10-tn1,1} "]"_{Bn1,1}
P_2: u_2 "[" (">"|"<")^{t1,2} (" ")^{10-t1,2} "]"_{B1,2} ... "[" (">"|"<")^{tn2,2} (" ")^{10-tn2,2} "]"_{Bn2,2}
 ⋮
P_m: u_m "[" (">"|"<")^{t1,1} (" ")^{10-t1,1} "]"_{B1,m} ... "[" (">"|"<")^{tnm,m} (" ")^{10-tnm,m} "]"_{Bnm,m}
```

donde

- `HH` es la hora del día simulado
- `MM` es el minuto del día simulado
- `m` es el número de paradas que tienen servicio
- `P_i` es el código de la i-ésima parada que tiene servicio
- u_i es el número de usuarios esperando en la i-ésima parada
- n_i es el número de autobuses actualmente efectuando recorridos desde o hacia la i-ésima parada
- $B_{j,i}$ es el j-ésimo autobús efectuando un recorrido desde o hacia la i-ésima parada
- $t_{j,i}$ es la posición del j-ésimo autobús efectuando un recorrido desde o hacia la i-ésima parada, medida en décimos
- “>” y “<” indican la dirección del autobús (“>” es de USB a la parada, “<” es de la parada a la USB)

Así, un estado de ejemplo del programa puede ser

```
7:34
BRT:   53 [>>>>       ] [>>>>>>>    ] [<<<<<<     ] [<<          ]
CHT:  179 [>>         ] [<<<<<<<<<< ]
BAR: 1023 [>>>>>>>>>> ]
CCH:   87 [>>>>>>>    ] [<<<<<<<<<  ] [<           ]
```

Representando un momento de la simulación en el que

- Son las 7:34 a.m.
- Hay 4 autobuses en la ruta de Baruta que llevan, respectivamente 40% y 70% del trayecto de ida; y a los otros dos que les resta el 60% y el 20% del trayecto de regreso
- Hay 2 autobuses en la ruta de Chacaito, uno lleva el 20% de ida y al otro le falta el 90% del regreso
- Hay 1 autobús en la ruta de Bellas Artes y lleva el 90% del camino de ida
- Hay 3 autobuses en la ruta a coche: uno lleva el 70% del camino de ida y los otros dos les falta el 80% y el 10% del camino de regreso

2.2 Formato de Archivos

2.2.1 Archivo de caracterización de la carga al sistema

El archivo de caracterización de la carga al sistema es un archivo separado por comas (CSV) en el siguiente formato:

```
Cod, Nombre, Recorr,  h1,  h2, ...,  h8
P_1,   PL_1, hh1:mm1, u1,1, u1,2, ..., u1,8
P_2,   PL_2, hh2:mm2, u2,1, u2,2, ..., u2,8
 ⋮
P_m,   PL_m, hhm:mmm, um,1, um,2, ..., um,8
```

donde

- `hk` es la k-ésima hora a la que arriban usuarios a la parada, indicada como un número entero (por ejemplo, si h1 es 7, el primer valor indica el número de usuarios que arriban a la parada a las 7:00)
- `PL_i` es el nombre largo de la i-ésima parada
- hh_i es la duración del recorrido entre la universidad y la i-ésima parada, en horas
- mm_i es la duración del recorrido entre la universidad y la i-ésima parada, en minutos adicionales a la hora
- $u_{i,k}$ es el número de usuarios que arriba a la i-ésima parada a ,a la k-ésima hora

2.2.2 Archivo de caracterización del servicio

El archivo de caracterización del servicio es un archivo con el siguiente formato:

```
P_1 hh1,1:mm1,1(c1,1) hh1,2:mm1,2(c1,2) ... hh1,n1:mm1,n1(c1,n1)
P_2 hh2,1:mm2,1(c2,1) hh2,2:mm2,2(c2,2) ... hh2,n2:mm2,n2(c2,n1)
 ⋮
P_m hhm,1:mmm,1(cm,1) hhm,2:mmm,2(cm,2) ... hhm,nm:mmm,nm(cm,nm)
```

donde

- $hh_{i,j}$ es la hora a la que parte el j-ésimo autobús de la universidad hacia la i-ésima parada
- $mm_{i,j}$ es los minutos después de la hora a los que parte el j-ésimo autobús hacia la i-ésima parada
- $c_{i,j}$ es la capacidad del j-ésimo autobús a la i-ésima parada

2.3 Funcionamiento del programa

El programa debe crear un proceso por cada una de las rutas. Cada proceso debe almacenar los datos de su ruta y mantener la hora simulada.

Al llegar la hora de cada autobús en el archivo de caracterización del servicio, el proceso debe crear un hilo que simulará a este autobús. El hilo debe mantener su estado actualizado, avanzando en el tiempo del recorrido un minuto por cada `t` segundos y reportando su nuevo estado al hilo principal de su proceso. El hilo principal debe enviar esta información al proceso padre, que se encargará de imprimir.

Cuando el autobús arriba a la parada (es decir, cuando el tiempo del recorrido se cumple) debe esperar 10×`t` segundos para simular la carga de pasajeros. El hilo principal, al recibir este reporte, debe restar la capacidad de este autobús del número de pasajeros esperando en la parada y contabilizar cuántos de ellos llegarán a tiempo (menos de 1½ horas después de haber llegado a la parada)

Una vez transcurrido el tiempo de simular la carga de pasajeros, el hilo ahora simula el recorrido de regreso, nuevamente avanzando en el tiempo del recorrido un minuto por cada `t` segundos.

Al arribar a la universidad (es decir, cuando el tiempo del recorrido de regreso se cumple), el hilo debe terminar. El hilo principal debe recibir el join respectivo y, una vez que no hay más buses programados, y todos los hilos han terminado, el proceso debe morir, pero no se debe perder la información de suficiencia de la ruta: al terminar la simulación, se debe imprimir el número de usuarios que llegó a tiempo para cada ruta, y el número que no

2.4 Sugerencias

- Establezca un pipe anónimo entre el proceso padre y cada uno de los procesos hijo
- Coloque la hora simulada en cada mensaje enviado por los pipe para poder detectar errores
- Utilice semáforos para sincronizar los relojes de los procesos cada hora (obligando a los procesos que alcancen a la siguiente hora primero, a esperar por los otros)
- En vez de esperar exactamente `t` segundos para sumar 1 al tiempo del recorrido, encuentre un múltiplo adecuado con el cual no haya diferencia en el funcionamiento
- Utilice `time_t` para las horas para aprovechar las rutinas de librería de manejo de horas
- Almacene el número de personas en la parada, no como un total, sino como una cola en donde cada elemento es las personas que llegaron a una hora (así, el primer elemento de la cola serían los usuarios que llegan a las 6:00 a.m., el segundo elemento, las que llegan a las 7:00 a.m. y así sucesivamente) y sólo elimine un elemento de la cola cuando su número de usuarios llegue a cero
    - Recorra la cola hasta la hora actual sumando los elementos para la impresión
- Cargue la caracterización de la carga en una estructura estática como un array o un struct
- Cargue la caracterización del servicio en una estructura dinámica como una lista enlazada

3 Informe

Debe escribir un informe que contenga

- Introducción, indicando la estructura del informe
- Decisiones de diseño e implementación, indicando las estructuras de datos u optimizaciones utilizadas, las dificultades encontradas y/o arreglos realizados sobre la marcha
- Evaluación de las configuraciones, incluyendo los resultados de correr con cada combinación de los archivos proporcionados y su análisis de los mismos
- Configuración propuesta, indicando y justificando cómo cree que se pueden mejorar las configuraciones dadas con al menos una corrida para soportar la propuesta. Observe, para este fin, que la universidad sólo cuenta con autobuses de 30, 45 y 60 puestos
- Conclusiones, recomendaciones y lecciones aprendidas

4 Requerimientos de la entrega

Todos sus códigos deben estar debidamente documentado y seguir las buenas prácticas de programación en Unix

El proyecto debe ser subido al Moodle de la materia en la sección marcada como “ Proyecto 1”. Sólo deberá efectuar una entrega por grupo.

5 Evaluación

El proyecto tiene una ponderación de 15puntos. Se asignarán

- 5 puntos por código
  - 1 punto por su manejo de procesos
  - 1 punto por su manejo de hilos
  - 1 punto por su comunicación entre procesos
  - 1 punto por su comunicación entre hilos
  - 1 punto por su manejo de la impresión y el programa principal
- 5 puntos por ejecución
  - 1 punto por el manejo de parámetros de cónsola
  - 1 punto por imprimir en el formato correcto
  - 1 punto por mantener el ritmo planteado (efectivamente actualiza cada `t` segundos)
  - 1 punto por incrementar y deincrementar correctamente el número de personas esperando en la parada
  - 1 punto por imprimir el resultado al final
- 5 puntos por informe
  - 1 punto por su introducción
  - 1 punto por sus decisiones de diseño
  - 1 punto por su evaluación de las configuraciones
  - 1 punto por su solución propuesta
  - 1 punto por sus conclusiones

El programa debe correr sin errores.

6 Puntos extra

Se asignarán puntos extra por permitir la opción de cambiar cualquiera de las suposiciones de la introducción de este enunciado. La opción debe poderse activar o desactivar con un parámetro de cónsola y debe estar explicada en el informe.