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

Prof. Fernando Torre Mora

Proyecto 1 Abril – Julio 2025

Sim Caracas

1 Introducción

Según cifras oficiales, la economía venezolana actualmente crece a una tasa del 9%. La última vez que creció a esta velocidad en 1991, hubo tal nivel de congestión de tráfico, que requirió la construcción de emergencia de una serie de elevados metálicos (o como se conocían en aquel entonces, "puentes de guerra") como se muestra en la Figura 1.

Figura 1. El elevado de la Avenida Roosevelt en La Bandera, uno de cuatro elevados metálicos notables de Caracas (los otros siendo el de la Avenida Urdaneta en La Candelaria, la Avenida Rio de Janeiro en Las Mercedes, y la Avenida Francisco de Miranda en Los Ruices). Fotografía VTV 2025

En preparación para este aumento vehicular, se le ha encomendado la tarea de desarrollar un simulador de tráfico sencillo que muestre diferentes zonas de la ciudad como nodos de un grafo; con las arterias viales que las conectan como aristas. Para simplificar, los nodos serán dispuestos en forma de cuadrícula, con un máximo de cuatro arterias viales posibles (norte, sur, este y oeste). Suponga un vehículo por residente.

Los simuladores de tráfico tienen dos problemas principales: la eficiencia de la simulación, y simular la demanda inducida. Para resolver este primer problema, se le pide usar proceso e hilos. Para resolver el segundo, se le pide simular el crecimiento de las zonas basándose en SimCity 1989.

La simulación se dividirá en días simulados, al final de los cuales se permitirá al usuario

- Agregar zonas
- Agregar arterias viales
- Ampliar arterias viales
- Guardar el grafo actual
- Salir del programa

2 Requerimientos del programa

Escriba un programa en C que pueda ser llamado desde la cónsola de Linux de la siguiente manera

```
$ ./simcaracas [-d <int>] [-t <float>] [-z <int>] [-f <archivo>]
```

Donde

- -d indica la duración del día medida en "ticks" del reloj simulado del programa (si no se pasa este parámetro, deberá suponerse por defecto un día de 64 ticks),
- -t indica la duración de un tick en segundos (por defecto 1.0) y
- -z indica el nivel máximo que pueden alcanzar las zonas (por defecto 10),
- -f es el archivo a cargar, si este existe (si no hay archivo, debe comenzarse con un grafo vacío)

2.1 Funcionamiento del programa

2.1.1 Procesos e hilos

El programa debe crear **un proceso para el manejo de las zonas y otro para el manejo del tráfico**. El primero de estos procesos debe asignar el manejo de cada zona a un hilo. El segundo, debe crear un hilo por cada conjunto de vehículos siguiendo una ruta (al cual llamaremos una "caravana").

Ambos procesos deben comunicarse con el proceso principal a través de *pipes* FIFO (no deben comunicarse entre sí). El proceso principal debe, como mínimo, tener los siguientes procesos hjo:

- un proceso para manejar la información que se envía y recibe a través de los *pipes*
- un proceso para llevar cuenta del tiempo (es decir, para decidir cuando avanza un tick del reloj simulado)
- un proceso para manejar la impresión por cónsola

El hilo que lleva cuenta del tiempo debe definir la duración de la mañana y la tarde como $\frac{d}{2}$ ticks y, cuando se alcanza este número de ticks, pedir al hilo que maneja los pipes que comunique a los procesos hijo el final de la mañana o de la tarde.

2.1.2 Zonas

Cada zona de la ciudad debe ser designada como fuente (hogares) ó sumidero (trabajos). A cada zona se le asigna un nivel que indica cuántos residentes o empleos existen en la zona (respectivamente). Para simplificar, el nivel será la potencia de dos de las personas que viven en la zona; de esta manera, las zonas fuentes comienzan en nivel 1 (dos personas residen ahí), y puede duplicar sus residentes si sube a nivel 2 (cuatro personas residen ahí).

Al crear o subir de nivel una zona fuente, debe encontrarse un sumidero disponible para cada uno de los nuevos residentes. A este fin, el hilo de este nodo realiza un recorrido en el grafo BFS: se almacenan los nodos vecinos en una cola (implementada como una lista enlazada) y se revisan en el orden agregado. Si uno de los nodos de la cola es sumidero y tiene puestos disponibles, se deben asignar a los residentes: se debe almacenar los nodos revisados para llegar desde la fuente a él como el trabajo de ese número de residentes (y, conversamente, debe marcarse como no disponibles esos puestos en el sumidero). Si los puestos son insuficientes para los residentes, se continúa revisando para encontrar trabajo a los restantes. Si se revisan todos los nodos, deben marcarse como desempleados.

Al crear o subir de nivel un nodo sumidero, debe realizarse un recorrido BFS, buscando, esta vez, los nodos con desempleados. Si se encuentra uno, debe asignársele el puesto al número de residentes desempleados de ese nodo (y, conversamente, debe almacenarse en el nodo la lista de nodos revisados – en orden inverso – como su ruta para ir al trabajo). Si se revisan todos los nodos, los puestos restantes deben marcarse como disponibles.

2.1.3 Tráfico

Para la simulación del tráfico, todos los hilos deberán esperar el inicio de un nuevo día, indicado por el proceso principal. En ese momento, se deberá crear un hilo por cada destino. Estos hilos asignarán, cada vez que avance un "tick" del reloj, todos los vehículos de esa ruta a la siguiente arista en la ruta, a menos que la arista esté por encima del 150% de su capacidad. Si se pueden asignar solo algunos de los vehículos a la arista, sin exceder la capacidad, deberá hacerse, pero en este caso, el hilo deberá almacenar en una lista enlazada en cuántas "caravanas" se ha separado su grupo inicial para que cada una pueda seguir avanzando acorde. Una vez que todas sus caravanas han llegado a su destino, el hilo debe morir.

Una vez que todos han llegado a su destino, debe simularse el viaje de retorno, creando nuevamente un hilo por cada ruta y asignando, con cada tick, los vehículos a cada arista. Si el proceso principal declara el final de la mañana, los vehículos que aún no han llegado a su destino se considera que "perdieron" el trabajo por no haber podido llegar, y deberá actualizarse el nodo fuente y sumidero acorde.

Una vez que todos han vuelto a su origen, se dice que ha terminado el "día" y debe indicarse la duración del trayecto de la mañana y el de la tarde, así como el nivel de empleo (porcentaje de puestos asignados a residentes) y desempleo (porcentaje de residentes sin puestos asignados) de la ciudad.

2.1.4 Demanda inducida

Al terminar el trayecto de la mañana, deberá sumarse tantos "puntos" a cada sumidero como puestos asignados tenga. Esto representa las ventas del día. Una vez que el nodo haya acumulado el cuadrado de su capacidad, subirá de nivel (así, un nodo nivel 3 tendrá 16 puestos y requerirá de 256 puntos para subir de nivel. Si solo 10 puestos están ocupados, requerirá de 26 días simulados para acumular los puntos requeridos, menos los puntos que ya acumuló en el nivel 2)

Al terminar el trayecto de la tarde, deberá sumarse tantos "puntos" a cada fuente como residentes empleados tenga. Esto representa los sueldos. Una vez que el nodo haya acumulado el cuadrado de su capacidad, subirá de nivel (así, un nodo nivel 3 tendrá 16 residentes y requerirá de 256 puntos para subir de nivel. Si solo 10 residentes están empleados, requerirá de 26 días simulados para acumular los puntos requeridos, menos los puntos que ya acumuló en el nivel 2)

2.1.5 Modificación del grafo

Al final de cada día simulado, el usuario tendrá la posibilidad de modificar el grafo agregando nodos y aristas. Cada nodo nuevo debe causar la creación respectiva de su hilo en el proceso de zonas, y estos deberán ejecutar su BFS respectivo.

2.2 I/O

Al inicio del programa debe mostrársele al usuario el grafo (si se cargó un archivo; si no, debe informársele que el grafo está vacío) y debe presentársele un menú que le permita Agregar zonas, Agregar arterias viales, Guardar el grafo actual o Salir del programa.

2.2.1 Salida esperada

Cada zona de la ciudad se representa con un código de tres letras. Para imprimir una zona, se deben mostrar 4 líneas: las cuales deben incluir

- Código de la zona
- Nivel de la zona
- Si es nodo fuente, cuántos desempleados hay; si es nodo sumidero, cuantos puestos disponibles hay
- Puntos acumulados

Adicionalmente, se debe indicar si las arterias viales conectadas están impidiendo que los vehículos que están en esa zona, salgan:

- Si la arteria vial está por encima del 50% de su capacidad (inclusive), se debe mostrar un signo de exclamación a su entrada. (si está por debajo, nada)
- Si está por encima del 100% de su capacidad (inclusive), se deben mostrar dos signos de exclamación.
- Si la arteria vial ha alcanzado el 150% de su capacidad, se deben mostrar tres signos de exclamación.

cada arista se representa con la cantidad de vehículos actualmente asignada a ella en el sentido que están asignadas. Así, si la zona CHT tiene todas las conexiones sobre capacidad, se puede mostrar de la siguiente manera

2.2.2

```
                    N1
                    CCB
                    1P
                !!! 1D
                150    150
  N9                N10 !!!             N8
  SAB     150 !!! CHT         150 !!! CHC
2530P !!! 150       2530P !!! 150     555P
  10D            !!! 10D               10D
                 150     150
                    N7   !!!
                    REC
                    10P
                    1D
```

Entrada esperada

Al final de cada día, debe presentársele al usuario el menú con las opciones para Agregar zonas, Agregar arterias viales, Ampliar arterias viales, Guardar el grafo actual o Salir del programa.

Para agregar una zona, el usuario debe especificar

- El código de tres letras de la zona
- El nivel inicial de la zona
- Cuál, de las zonas existentes, es la más cercana
- Cómo llegar, desde esta zona más cercana, a la nueva (norte, sur, este u oeste)

Así, si el usuario especifica la zona nueva PTE nivel 9 al este de CAL, debe crearse un nuevo nodo en el grafo y una arista de CAL a PTE. Debe marcarse en CAL, que su conexión este está ocupada, y en PTE que su conexión oeste está ocupada. Esto asegurará que, si el usuario intenta agregar una segunda zona al

este de CAL (que sobreescribiría PTE) o una zona al oeste de PTE (que sobreescribiría CAL), el programa debe imprimir un mensaje de error. El programa no se debe cerrar por este error.

Para agregar o ampliar una vía, sólo deben especificarse las dos zonas entre las cuales se encuentra la vía. Las vías siempre comienzan con una capacidad predeterminada de 100 vehículos por tick en cada dirección, y ampliarlas siempre duplica esta capacidad. Agregar una vía debe marcar como ocupada la conexión en los nodos respectivos (no pueden haber dos vías ocupando la misma conexión). Todas las vías son bidireccionales.

"Guardar el grafo" requiere que el usuario ingrese el nombre del archivo.

"Salir del programa" debe terminar todos los hilos y procesos correctamente.

2.3 Formato de archivo

Los archivos de entrada deben estar en formato CSV donde las columnas deben ser

- Nombre de la zona identificada por su código de tres letras
- Nivel de la zona
- Número de puntos acumulados
- Nombre de la zona conectada al norte
- Capacidad de la vía norte
- Nombre de la zona conectada al sur
- Capacidad de la vía sur
- Nombre de la zona conectada al este
- Capacidad de la vía este
- Nombre de la zona conectada al oeste
- Capacidad de la vía oeste

Nótese que la primera línea de cualquier archivo CSV siempre debe ser los nombres de las columnas.

Se proporcionará un archivo de ejemplo con un plano significado de Caracas.

2.4 Sugerencias

Diseñe y pruebe sus estructuras de datos primero, la comunicación entre procesos para poblarlas después, luego el manejo de hilos y finalmente el funcionamiento general. Defina impresiones para cada parte del programa a fin de probar la concurrencia antes de diseñar la impresión en el formato pedido.

3 Informe

Debe escribir un informe que contenga

- Introducción, indicando motivación y estructura del informe
- Decisiones de diseño e implementación, dando la estructura del programa y las dificultades encontradas
- Evaluación del archivo dado, indicando cuáles zonas, en sus pruebas, se congestionaban lo suficiente para ameritar un "elevado de emergencia" (no razonablemente arreglables ampliando la vía)

- Conclusiones, resumiendo el proyecto, su estado actual, las lecciones aprendidas y dando recomendaciones

4 Requerimientos de la entrega

Todos sus códigos deben estar debidamente documentados y seguir las buenas prácticas de programación en Unix incluyendo un *makefile*. Deberá entregar un archivo comprimido con su código y su informe. Sólo deberá efectuar una entrega por grupo.

5 Evaluación

El proyecto tiene una ponderación de 15 puntos. 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
-
- 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.