Universidad Nacional Autónoma de México

Facultad de Ingeniería

Profesor(a): Gunnar Eyal Wolf Iszaevich

Semestre 2024-1

Proyecto No. 2: Una situación cotidiana paralelizable

Sistemas Operativos

Grupo: 6

Alumno: Vázquez Reyes Sebastián (318150923)

Fecha: 6 de noviembre de 2023

Una situación cotidiana paralelizable

Identificación y descripción del problema

1. Descripción

Hace un año aproximadamente, surgió una polémica con TicketMaster y uno de los cantantes mas populares entre la chaviza: Bad Bunny. Resulta que TicketMaster vendió más boletos de los que en verdad estaban disponibles debido a la alta demanda y tráfico en su plataforma, provocando un problema enorme en la fecha del concierto. Sumado a la reventa de boletos falsos, hubo muchísimas personas que, a pesar de obtener un boleto de forma legal, no tuvieron acceso al concierto.

Entonces, resulta obvio el problema de la concurrencia en este tipo de eventos donde hay altos volúmenes de gente, la cosa esta en controlar el numero de boletos existentes para que el sistema sepa cuando puede seguir vendiendo y cuando no.

2. ¿Dónde pueden verse las consecuencias nocivas de la concurrencia? ¿Qué eventos pueden ocurrir que queramos controlar?

De entrada, el manejo erróneo de la concurrencia provocó una enorme cantidad de personas que prácticamente regalo su dinero. TicketMaster posteriormente tuvo que devolver una cantidad a cada damnificado que suman millones de pesos. Además, un escándalo de esta magnitud mancha la imagen del único sitio mayormente conocido para la compra-venta de boletos en México.

Los eventos a controlar serian: el acceso al sitio de compra de parte de varios usuarios y el acceso al pago del o los boletos, una vez que se hayan confirmado todos los datos de compra.

3. ¿Hay eventos concurrentes para los cuales el ordenamiento relativo no resulta importante?

Para este caso en particular, no encontré ningún evento en el que sea necesario el ordenamiento, puesto que los usuarios accederán al sistema y pagaran en función de la velocidad de su internet y computador. Son variables que no podemos controlar.

Implementación del modelo

- **Descripción de los mecanismos de sincronización empleados**

Para la solución del problema utilicé un semáforo y un MUTEX. El semáforo se encarga de dejar pasar un numero determinado de personas al sitio de compra, similar al mundo real, donde ocurre algo parecido cuando hay un gran numero de personas consultando un sitio a la vez. Este semáforo esta inicializado en 12, para mantener un numero pequeño de hilos en la ejecución y no tener un resultado de ejecución tan grande.

El MUTEX se encarga de ceder el paso a una sola persona para pagar el boleto, actualizar su saldo y actualizar los datos de existencia de los boletos.

Además, también uso un MUTEX para que los hilos accedan de forma ordenada a una función que les permite imprimir sus mensajes en la pantalla, para que se vea mas ordenada la pantalla de ejecución.

- **Lógica de operación**

El código presenta una sección compartida en el número de boletos disponibles. Cada hilo es un individuo con ganas de comprase un boletito, si alguien compra un boleto, afectará el stock y los demás tendrán menos oportunidades de comprarlos.

Tras la creación de los hilos, cada uno muestra en pantalla su “id” y el concierto al que desea asistir. Luego, cada uno de ellos toma un descanso por una cantidad de tiempo escogida al azar y posteriormente invocan a la función “compraBoleto(id, tarjeta, con)”.

La función recibe como parámetros el id del hilo/cliente, su dinero guardado y el concierto al que desea asistir. En esta función se lleva a cabo la recolección de la información de compra de los hilos, antes de poder pasar a pagar su o sus boletos, primero escogen la cantidad que quieren comprar y se corrobora que: 1-. Existan suficientes boletos para que puedan comprar los que desean, 2-. Tengan suficientes fondos monetarios en su tarjeta de débito. Al acceder a esta función, ocupan un lugar del semáforo, por lo que hay hilos que deben esperar a que la operación de otro hilo acabe y puedan acceder a comprar un boleto.

Si el hilo cumple con los requisitos de compra, manda a llamar a la función “pagarBoleto(tarjeta, precio, con, boletos)”. Esta función recibe como parámetros el

dinerito guardado de la persona/hilo, el precio total de los boletos que desea, el concierto al que desea asistir y la cantidad de boletos que compra. Tras realizar la transacción, el hilo sale de esta función y regresa a compraBoleto para avisar de su operación. Luego sale del semáforo y de la cola de compra.

Los hilos deben esperar a que sus compañeros salgan del semáforo para poder cargar sus datos de compra, y deben esperar que salgan del MUTEX para poder realizar la transacción.

- **Descripción del Entorno de Operación**

Para el desarrollo de mi programa utilice:

- Python 3.9.13
- Las bibliotecas: time, random y threading
- El sistema operativo Windows 10