Universidad Nacional Autónoma de México

Materia: Sistemas Operativos

Profesor: Gunnar Eyal Wolf Iszaevich

Tenorio Martinez Jesus Alejandro

Tarea 1: Ejercicio de sincronización

El problema que se eligió fue el siguiente:

Un profesor de la facultad asesora a varios estudiantes, y estamos en su horario de atención. Modelar la interacción durante este horario de modo que la espera (para todos) sea tan corta como sea posible Un profesor tiene x sillas en su cubículo Cuando no hay alumnos que atender, las sillas sirven como sofá, y el profesor se acuesta a dormir la siesta. Los alumnos pueden tocar a su puerta en cualquier momento, pero no pueden entrar más de x alumnos Para evitar confundir al profesor, sólo un alumno puede presentar su duda (y esperar a su respuesta) al mismo tiempo. Los demás alumnos sentados deben esperar pacientemente su turno. Cada alumno puede preguntar desde 1 y hasta y preguntas (permitiendo que los demás alumnos pregunten entre una y otra)

Explicación Código propuesto en C

El código simula que Gunnar que atiende a varios estudiantes, permitiendo que solo tres alumnos pasen al cubículo a la vez, con la condición de que el profesor solo pueda atender a un alumno a la vez. Para gestionar esta situación, el código utiliza la librería OpenMP para paralelismo, con el fin de simular múltiples alumnos (hilos) que compiten por la atención del profesor.

En el código cada hilo representa un alumno, todos los alumnos pueden hacer una pregunta a la vez

1. **Semáforos para controlar el acceso**:
    - Las variables semaforo y semaforo2 se usan para limitar el acceso de los alumnos.
        - semaforo controla cuántos alumnos pueden entrar al cubículo (un máximo de 3).
        - semaforo2 asegura que Gunnar atienda a un alumno a la vez, restando el valor por cada alumno atendido.
2. **Inicio de paralelismo**:
    - El pragma #pragma omp parallel num_threads(x) crea tantos hilos como alumnos cada hilo actuando como un estudiante.
    - El pragma #pragma omp single asegura que solo un hilo (un alumno) imprimirá el mensaje de que Gunnar ha despertado cuando hay alumnos con dudas.
3. **Control de acceso de los alumnos (semaforo)**:
    - Cada alumno intenta entrar al cubículo verificando el valor de semaforo. Si hay lugares disponibles (más de 0), decrementa semaforo y continúa. De lo contrario, el alumno queda esperando.
4. **Atención crítica del profesor**:

- El bloque crítico #pragma omp critical asegura que solo un hilo (un alumno) pueda ser atendido por el profesor a la vez.
- Dentro de este bloque:
  - Se decrementa semaforo2 (que controla el número de alumnos atendidos).
  - El mensaje informa que Gunnar está atendiendo la duda de un alumno en particular (identificado por el número de hilo).
  - Después de atender al alumno, el semáforo se restaura cuando tres alumnos han sido atendidos (si semaforo2 llega a 0, lo que indica que se ha completado una ronda).
- Los demás hilos (alumnos) deben esperar hasta que Gunnar termine con el alumno actual.

5. **Rondas de atención**:
   - Cuando Gunnar termina de atender a 3 alumnos (semaforo2 == 0), los semáforos se reinician para permitir el paso de otros 3 alumnos, simulando la siguiente ronda de atención.
6. **Barriera y mensaje final**:
   - #pragma omp barrier se usa para sincronizar los hilos. Todos los alumnos deben llegar a este punto antes de continuar.
   - Finalmente, el pragma #pragma omp single se asegura de que solo un alumno imprima el mensaje indicando que no quedan más dudas y que Gunnar puede volver a dormir.