Compatibilidad de semáforos e hilos en el problema del productor-consumidor

Asier Cabo Lodeiro y Hugo Gilsanz

1. Introducción

Siguiendo el esquema de los anteriores informes, el problema del productor-consumidor es un ejemplo clásico de sincronización en sistemas concurrentes (véase *Problema del productor-consumidor con memoria compartida*). Para evitar condiciones de carrera y garantizar un acceso seguro a la memoria compartida, se emplean mecanismos de sincronización como los semáforos (véase *Uso de semáforos para la resolución de carreras críticas*). En este informe, se presenta una solución basada en hilos (threads) en lugar de procesos, utilizando la biblioteca pthread.

2. Problemática y posible solución

2.1 Descripción del problema

En esta implementación, dos hilos operan simultáneamente como productor y consumidor y deben coordinarse para evitar la sobreescritura y el acceso a posiciones vacías del buffer, así como la aparición de carreras críticas.

2.2 Incorporación de semáforos

Al igual que en la solución del informe anterior, se implementan tres semáforos.

- **VACIAS**: controla la cantidad de espacios disponibles en el buffer.
- **LLENAS**: indica la cantidad de elementos disponibles para ser consumidos.
- **MUTEX**: garantiza la exclusión mutua en la sección crítica.

2.3 Incorporación de hilos

En lugar de procesos, la solución se basa en dos hilos creados con pthread_create, uno para el productor y otro para el consumidor. No es necesario explicar mucho más pues el papel de los hilos es sustituir la generación de dos procesos al crearse hilos que trabajarán de forma simultánea hasta su fin (bucles limitados).

3. Estructura y Análisis del código

Como se están empleando como base los códigos del apartado anterior (véase el informe *Problema del productor-consumidor con memoria compartida*) el código resultante juntará todas las funciones previamente descritas para el tratado de caracteres del buffer, además de modularizar la acción de productor y consumidor en funciones homónimas que los hilos puedan llamar para operar de forma simultánea.

3.1 Implementación en C

Se implementa en C utilizando la biblioteca pthread.h para la creación y sincronización de hilos, semaphore.h para el uso de semáforos. Se definen las estructuras de datos, se inicializan los semáforos y se gestiona la memoria compartida adecuadamente dentro del propio programa.

La ejecución del programa consiste en:

1. Inicializar la memoria compartida y los semáforos.
2. Crear los hilos productor y consumidor.
3. Esperar la finalización de ambos hilos con pthread_join().
4. Destruir los semáforos al finalizar la ejecución.

4. Instrucciones de Compilación y Ejecución

5.1 Compilación

Para compilar el código, se puede usar el compilador GCC de la misma manera que anteriormente, pero siendo ahora imprescindible incluir la biblioteca de hilos (-pthread). Ejemplo de compilación: gcc op1.c -o op1 -lrt -pthread

5.2 Ejecución

En una terminal debe ejecutar el programa sin argumentos: ./op1

5. Conclusiones

La implementación con hilos y semáforos permite una ejecución segura y coordinada del problema productor-consumidor. Gracias al uso de semáforos, se evitan problemas de condiciones de carrera y se logra una comunicación eficiente entre los hilos. Además, la introducción de pausas aleatorias simula un comportamiento más realista en entornos de producción y consumo concurrentes.