SISTEMAS CONCURRENTES Y DISTRIBUIDOS

Álvaro Fernández García

-Documentación productor-consumidor-

-Hebras utilizadas:

Se implementan únicamente dos hebras, cada una de ellas realizará una de las tareas. Por un lado tenemos la hebra_productora que ejecutará la función productor (sin ningún parámetro) y por otro la hebra_consumidora que ejecutará la función consumidor (también sin ningún parámetro).

-Implementación decidida para el acceso al vector:

Se ha optado para esta implementación por la cola de tipo LIFO (pila acotada) en la cual se emplea una varible llamada primera_libre que contiene la primera posición libre del vector en la cual se puede leer y escribir. Esta variable se incrementará al escribir y por el contrario se decrementará al leer. Sigue un mecanismo “último en entrar, primero en salir”.

-Variables empleadas:

Las principales variables que se han utilizado son:
- Vector buffer: se trata del vector en el cual se almacenarán los elementos insertados por el productor y de donde el consumidor extraerá los elementos correspondientes. Se trata por tanto de un recurso compartido al que intentarán acceder ambas hebras, en consecuencia será necesario controlarlo en exclusión mútua (se verá en apartados posteriores).
-Número de items: se trata de una constante que establece el número de veces que el productor introduce y que el consumidor extrae un dato.
-Tamaño del vector: cantidad de elementos que soporta el buffer
-Semáforos necesarios: Se expecificará en el apartado siguiente.

-Semáforos:

Se utilizarán tres semáforos:
-Puede leer: se trata de un semáforo que controla cuando el consumidor puede extraer un dato. Su inicialización será por tanto a 0, puesto que hasta que no haya accedido por primera vez el productor al buffer, el consumidor no podrá consumir ningún elemento. Presenta un wait al principio de la función consumidor y recibirá un signal situado al final de la función productor, una vez introducido el dato.
-Puede escribir: Este semáforo le indica al productor cuando hay espacio disponible en el buffer para introducir un elemento. Su inicialización será al valor que tenga el tamaño del buffer, de esta forma conseguimos que cada vez

que introduzca un elemento y pase por el wait su valor se decremente en uno (por tanto tendrá un wait al principio de la función productor). Si este llega a cero, significará que no queda más espacio libre en el buffer y por tanto no puede escribir. En contrapartida cuando el consumidor extrae un dato este deberá mandar un signal, o bien para desbloquear al productor, o bien para incrementar su valor, indicando que hay más huecos dispobles, (por tanto este semáforo tendrá un signal al final de la función consumidor).
-Mutex: se trata de un semáforo para controlar la exclusión mútua el buffer, al tratarse de un semáforo de exclusión mutua se inicializa a 1, y presentará un wait antes de acceder al vector y un signal después en cada una de las funciones.

-Documentación fumadores-

-Hebras utilizadas:

En este proceso se utilizarán cuatro hebras: una de ellas representará al estanquero mientras que las tres restantes representará a cada uno de los fumadores. La primera ejecutará la función estanquero, mientras que las tres últimas ejecutarán la función fumador (al tratarse de tres hebras que realizan la misma función pero con distintos parámetros utilizaremos un vector y un bucle for para su tratamiento.

-Variables:

Además de los semáforos (los cuales se explicarán en el apartado siguiente), solo se utiliza una variable global para indicar el número de fumadores que tenemos en nuestro programa.

-Semáforos utilizados:

-Mostrador: este semáforo representa el mostrador del estanco, en él el estanquero coloca el recurso que necesitan los fumadores. Se inicializará a 1, puesto que el estanquero tiene que colocar en primer lugar un recurso y solo uno (no puede haber más recursos en el mostrador), por tanto presentará un wait al principio de la función estanquero y antes de colocar ningún recurso y un signal en la función fumador que le indique al estanquero cuando el fumador ha recogido su recurso.
-Semáforos de fumadores: los semáforos de los fumadores, debido a que tienen la misma estructura, se han almacenado en un vector. Estos semáforos le indican a los fumadores cuando el estanquero ha producido el recurso que necesitan para fumar. Todos se inicializarán a 0, puesto que hasta que el estanquero no haya colocado un recurso ninguno de ellos podrá fumar. Presentará un wait al principio de la función fumador, y un signal en la función estanquero cuando este haya producido el recurso necesario (obviamente el signal y el wait se realizará a través del índice al fumador que le corresponde

no se realizará a todos).
-Mutex: como su nombre indica se trata de un semáforo para implementar la exclusión mutua , cómo se ha explicado en el apartado anterior. Realmente este semáforo es ajeno a la implementación del problema de los fumadores y simplemente se ha añadido para evitar que las distintas hebras intenten escribir al mismo tiempo por la salida estándar.