SCD – Practica 3

Jorge Gangoso Klöck 49398653N

Implementación de algoritmos distribuídos con MPI.

1-Productor-Consumidor con varios Productores y Consumidores

Para esta solución creamos dos variables enteras constantes al inicio del programa que representen las etiquetas de productor y consumidor.

Cuando los productores quieran enviar un valor al buffer deberán hacerlo identificandose con la etiqueta de productor y de igual manera los consumidores para recibir.

Por tanto ya sólo es necesario determinar el comportamiento del buffer, el cual, cuando el buffer esté lleno sólo aceptará mensajes con la etiqueta de consumidor. Cuando esté vacío sólo con la etiqueta de productor y cuando esté en cualquier estado intermedio recibirá cualquier tipo de petición.

Por ello, en el tratamiento de mensaje se debe revisar mediante status.MPI_TAG si el mensaje recibido fue de un productor o de un consumidor y realizar la acción pertinente.

2-Filósofos con Interbloqueo

Para esta solución sólo era necesario completar la forma en que se solicitaban y liberaban los tenedores así como el comportamiento de los tenedores cuando son reclamados y liberados.

Ésto se realiza mediante envíos síncronos seguros utilizando los identificadores de los procesos para determinar que tenedores debe reclamar cada filósofo y qué filósofo es el que debe liberar un tenedor ocupado.

Este sistema sin embargo puede llevar a interbloqueo si por ejemplo todos los filósofos cogen su tenedor izquierdo, antes de que cualquiera de ellos llegue a coger el derecho, en cuyo caso todos esperan a que se libere un tenedor pero ninguno libera su tenedor.

3-Filósofos Solucionados

El sistema más simple para evitar el interbloqueo es cambiar el orden en que uno de los filósofos (el primero en mi ejemplo) solicita sus tenedores, de ésta manera no se puede llegar a la situación que declaramos antes ya que siempre habrá al menos uno que pueda coger ambos cubiertos.

En cuanto al código es cuestión de añadir una pequeña condición y si el filósofo es el número 0, cambia el orden en que coge sus cubiertos.

4-Filósofos con Camarero

Esta última solución trata con el interbloqueo impidiendo que todos los filósofos intenten coger sus cubiertos al mismo tiempo, ya que para pedir sus cubiertos deberán sentarse primero, y el camarero no permitirá que todos se sienten al mismo tiempo.

A nivel de programación apenas son necesarios algunos cambios en la solución 2 con interbloqueo.

Ajustamos las variables para un nuevo proceso, le asignamos el numero 10 y añadimos dos envíos síncronos seguros en el proceso de los filósofos donde antes de pedir los cubiertos deben enviar una

petición de sentarse al camarero y otro mensaje para avisar de que se levanta tras liberar sus cubiertos.

El proceso del camarero es sencillo también. En una variable recuerda el número de filósofos sentados y si sólo falta uno para que estén todos sentados se niega a recibir mensajes para sentarse. En cualquier otra situación utiliza los mensajes para sentarse o levantarse para llevar correctamente la cuenta de los filósofos que se encuentran sentados en cada momento.