ELVI MIHAI SABAU SABAU - 51254875L - 15/11/2022

Programacion Concurrente - Practica 4

Algoritmos con espera ocupada (Parte 2)

Algoritmo de Dekker (C/Dekker.c)

Después de ejecutar el código durante varias iteraciones, vemos que el resultado es el esperado y que funciona correctamente. Esto se debe a que el algoritmo asegura la exclusión mutua, cumple con la condición de progreso y satisface la limitación de espera. El unico inconveniente de este algoritmo es que es dificil de escalar a más procesos.

Algoritmo de Peterson (Python/Peterson.c)

El algoritmo de Peterson es una simplificación del algoritmo de Dekker, funciona correctamente y asegura la exclusión mutua al igual que el de Dekker, la diferencia con el algoritmo de Dekker es que en el algoritmo de Peterson es más sencillo y ahorra varios ciclos de procesamiento permitiendo asi el escalado.

Algoritmo de Hyman (Java/Hyman.java)

Tras varias ejecuciones el programa devuelve el resultado esperado, pero el incremento de `n` solo se produce en un hilo. Sin embargo, si no ejecutamos el código con `taskset` si que se ejecutan ambos hilos. Con takset también es posible que se ejecuten ambos hilos especificando varios nucleos al momento de arrancar el programa.

Algoritmo Lamport (Python/Lamport.py)

Al ejecutarlo con 4 hilos, (nosotros lo hemos ejecutado con 4 hilos, pero este algoritmo se puede implementar para N procesos) tras varias ejecuciones el programa devuelve el resultado esperado cumpliendo los requisitos para la exclusión mutua.