PRACTICAS DE PROGRAMACIÓN CONCURRENTE 2020/2021. Sesión 3

Programación Concurrente

27 de octubre de 2020

Algoritmos con espera ocupada

1. Probar con hilos POSIX (subapartados a y b), con hilos Java (subapartados b y c) y con hilos Python (c y d) los siguientes algoritmos vistos en clase que utilizan la espera ocupada (algoritmos no eficientes) para la exclusión mutua:

   a) Primer intento (alternancia)

   b) Segundo intento (falta de exclusión mutua)

   c) Tercer intento (posible interbloqueo)

   d) Cuarto intento (espera infinita)

2. Según la ejecución de cada uno de los algoritmos, anota en tu memoria de prácticas los fallos detectados en cada uno. Si no observas la salida esperada prueba a modificar los tiempos de respuesta de tus hilos para obtenerla si fuera posible. A la hora de estudiar el comportamiento de los hilos en la sección crítica ten en cuenta que, en Java, las asignaciones de tipos primitivos se realiza de forma atómica. No olvides comentar tu código.

3. Ejemplo código en C para el primer intento:

```
/**
   $hilos
   Compilación: cc -o hilos hilos.c -lpthread
**/
#include
#include
#include
#include
```

```
<pthread.h>
<stdio.h>
<string.h>
<stdlib.h>
int I = 0;
int turno = 1;
void *codigo_del_hilo (void *id){
  int i = *(int *)id;
  int j = (i == 1)? 2 : 1;
  int k;
  for(k=0; k<100; k++){
    // protocolo de entrada
    while(turno == j);
    // Sección crítica
    I = (I + 1)%10;
    printf("En hilo %d, I=%d\n", i,I);
    // protocolo salida
    turno = j;
    // Resto
  }
  pthread_exit (id);
}
int main(){
  int h;pthread_t hilos[2];
  int id[2]={1,2};
  int error;
  int *salida;
  for(h=0; h<2; h++){
    error = pthread_create( &hilos[h], NULL, codigo_del_hilo, &id[h]);
    if (error){
      fprintf (stderr, "Error: %d: %s\n", error, strerror (error));
      exit(-1);
    }
  }
  for(h =0; h < 2; h++){
    error = pthread_join(hilos[h], (void **)&salida);
    if (error)
      fprintf (stderr, "Error: %d: %s\n", error, strerror (error));
    else
      printf ("Hilo %d terminado\n", *salida);
  }
}
```

4. Ejemplo código en Java para el segundo intento:

```
import java.lang.Math; // para random
public class Intento2 extends Thread {
  static int n = 1;
  static volatile int C[] = {0,0};
  int id1; // identificador del hilo
  int id2; // identificador del otro hilo

  public void run() {
    try {
      for(;;){
        while(C[id2] == 1);
```

```
        C[id1] = 1;
        sleep((long)(100*Math.random()));
        n = n + 1;
        System.out.println("En hilo "+id1+", n = "+n);
        C[id1] = 0;;
      }
    }
    catch( InterruptedException e ){return;}
  }

  Intento2(int id) {
    this.id1 = id;
    this.id2 = (id == 1)? 0 : 1;
  }

  public static void main(String args[]) {
    Thread thr1 = new Intento2(0);
    Thread thr2 = new Intento2(1);

    thr1.start();
    thr2.start();
  }
}
```

NOTA: observa el atributo volatile para la variable turno, ¿qué ocurrre si lo eliminas?

5. Ejemplo código en Python para el tercer intento:

```
import threading

states = [False, False]

THREADS = 2
MAX_COUNT = 100000

counter = 0

def entry_critical_section(i):
    global states
    states[i] = True
    while states[(i+1)%2]:
      pass

def critical_section(i):
    global counter
    counter += 1

def exit_critical_section(i):
    global states
    states[i] = False

def thread(i):
    for j in range(MAX_COUNT//THREADS):
        entry_critical_section(i)
        critical_section(i)
```

```
        exit_critical_section(i)

def main():
    threads = []
    for i in range(THREADS):
        # Create new threads
        t = threading.Thread(target=thread, args=(i,))
        threads.append(t)
        t.start() # start the thread

    # Wait for all threads to complete
    for t in threads:
        t.join()

    print("Counter value: {} Expected: {}\n".format(counter, MAX_COUNT))


if __name__ == "__main__":
    main()
```

Entrega

- Adjunta a una tutoría en UA-Cloud hasta el dia 6 de noviembre.