Contenidos

- El problema de la sección crítica
- Semáforos
- **Regiones críticas**
- Monitores

Bibliografía

- Programación Concurrente
  - J. Palma, C. Garrido, F. Sánchez, A. Quesada, 2003
  - Capítulo 5
- Concurrent Programming
  - A. Burns, G. Davies. Addison-Wesley, 1993
  - Capítulo 7
- Sistemas Operativos
  - A. Silberschatz, P. Galvin. Addison-Wesley, 1999
  - Capítulo 6

Región crítica (RC)

- Brinch Hansen, 1972

```
var
   i,j: integer;
   resource V: i, j;
...
...
region V do
   S;
```


RC

- La semántica de la RG establece
  - Los procesos concurrentes sólo pueden acceder a las variables compartidas dentro de su correspondientes RC
  - Un proceso que quiera entrar a una RC lo hará en un tiempo finito
  - En un instante *t*, sólo un proceso puede estar dentro de una RC determinada
  - Un proceso está dentro de una RC un tiempo finito, al cabo del cual la abandona

RC

- Esto significa que:
  - Si el número de procesos dentro de una RC es igual a 0, un proceso que lo desee puede entrar a dicha RC
  - Si el número de procesos dentro de una RC es igual a 1 y N procesos quieren entrar, esos N procesos deben esperar
  - Cuando un proceso sale de una RC se permite que entre uno de los procesos que esperan

RC

- Las decisiones de quien entra y cuando se abandona una RC se toman en un tiempo finito
- Se supone que la puesta en cola de espera es justa

Región crítica condicional (RCC)

```
Var
   i,j: integer;
   resource V:i,j;
...
...
region V when B do
   S;
```


RCC

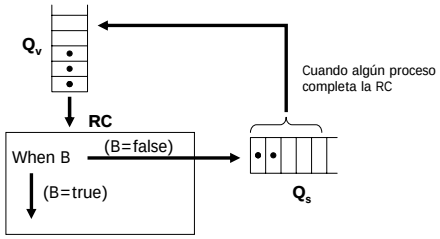
- Su funcionamiento es similar a la RC sólo que además para que un proceso ejecute S, la condición B debe ser cierta
- La evaluación de la condición B se considera parte de la región crítica
- En caso de evaluarse a falso, abandona la RC para permitir a otros procesos entrar en ella

RCC

- Un proceso que haya evaluado la condición a falso no vuelve a entrar a su RC hasta que otro proceso abandone ésta (espera pasiva)
  - Se vuelve a ejecutar cuando ¡posiblemente! alguien haya modificado dicha condición

Implementación

- Brinch Hansen sugirió el empleo de dos colas para la implementación de las RCC



Limitaciones de las RCC's

- Aunque mejoran algunos aspectos negativos de los semáforos, tienen algunas limitaciones:
  - Pueden aparecer a lo largo de todo el programa
  - No se garantiza la integridad de las estructuras de datos compartidas
  - Realizar una implementación eficiente de las mismas es una tarea difícil

Ejercicios

- En un sistema concurrente existen dos tipos de procesos, A y B, que compiten por utilizar cierto recurso. Al recurso se accede mediante la rutina de solicitarlo, esperar hasta que se pueda usar, usarlo y luego liberarlo. En cualquier momento puede haber un máximo de N procesos de cualquier tipo usando el recurso (N es constante). Por otro lado, para que un proceso de tipo A pueda entrar a emplear el recurso, debe haber al menos el doble de procesos de tipo B que procesos de tipo A dentro del recurso. Diseñe un algoritmo que cumpla con estas reglas de funcionamiento empleando regiones críticas

Ejercicios

- Supongamos dos operaciones: pedir_memoria(cantidad) y dejar_memoria(cantidad). Cuando un proceso pide memoria, se bloquea hasta que haya la cantidad pedida de páginas libres, una vez que ocurre las toma. Un proceso devuelve las páginas llamando a dejar_memoria. Implementar dichas operaciones usando RCC en los siguientes supuestos:
  - Implementar la sincronización sin establecer ninguna política de quién recibe primero.
  - Modificar lo anterior para que el trabajo que pide menos memoria sea el primero.
  - Cambiar la política anterior para que el primero en llegar sea el primero en salir o ser atendido.
  - Suponer que pedir y dejar devuelven páginas contiguas. Es decir, si un proceso reclama dos páginas, se retrasa hasta que haya dos páginas adyacentes disponibles. Desarrollar implementaciones de esa versión de pedir_memira y dejar_memoria. Elegir una representación del estado de páginas de memoria.