Tema 3: Cerrojos y Barreras (Parte 2: Barreras)

Elvira Albert

Departamento de Sistemas Informáticos y Computación

Universidad Complutense de Madrid

elvira@sip.ucm.es

Madrid, Marzo, 2021

3.4 Sincronización con Barreras

- Los algoritmos paralelos iterativos típicamente iteran sobre partes de los datos y una iteración puede depender de la anterior
- Podríamos implementarlo utilizando co:

```
while (true) {
  co[i=1 to n]
    código iteración i
  oc;
}
```

- Poniendo un punto de parada (barrera):

```
process Worker[i=1 to n]
while (true) {
   código iteración i
   barrera terminación "n" tareas
}
```

3.4 Sincronización con Barreras

Contadores compartidos

- Utilizando una variable compartida "counter"

  <await (count == n) >
- Tiene varios problemas:
  - cómo poner el contador a 0?
  - inc tienen que ser atómicos
  - contención de memoria en acceso a count

Banderas y coordinadores

- Utilizando n variables que suman el mismo valor

  <await (count[1]+...+count[n] == n) >
- reduce contención de memoria si count[i] en caches distintas
- cómo se implementa el await y el reseteo?
  - con un coordinador y variables adicionales
  - cada worker es también coordinador

Barreras simétricas

- una barrera simétrica de n procesos se construye a partir de barreras simétricas de 2 procesos
- esquemas de interconexión:
  - barrera mariposa
  - barrera diseminación
- evitar condiciones de carrera

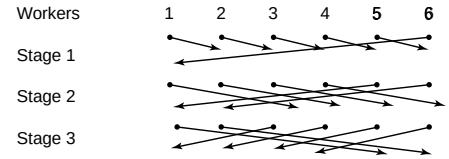

Figure 3.16 Dissemination barrier for 6 processes.

3.5 Algoritmos de datos paralelos

Objetivos

- conocer técnicas básicas usadas en algoritmos de datos paralelos
- uso de barreras de sincronización

Ejemplos

- Computaciones de prefijos paralelos
- Operaciones sobre listas enlazadas
- Computaciones Grid
- Computación paralela con bolsa de tareas