2º curso / 2º cuatr.

Grado en Ing. Informática

Arquitectura de Computadores
Tema 3

Lección 9. Consistencia del sistema de memoria

Material elaborado por los profesores responsables de la asignatura:
Mancia Anguita – Julio Ortega

ugr | Universidad de Granada

ETSIIT
Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación

ATC
Departamento de Arquitectura y Tecnología de Computadores
UNIVERSIDAD DE GRANADA

Lecciones

- Lección 7. Arquitecturas TLP
- Lección 8. Coherencia del sistema de memoria
- Lección 9. Consistencia del sistema de memoria
  - Concepto de consistencia de memoria
  - Consistencia secuencial
  - Modelos de consistencia relajados
- Lección 10. Sincronización

Objetivos Lección 9

- Explicar el concepto de consistencia.
- Distinguir entre coherencia y consistencia.
- Distinguir entre el modelo de consistencia secuencial y los modelos relajados.
- Distinguir entre los diferentes modelos de consistencia relajados.

Bibliografía Lección 9

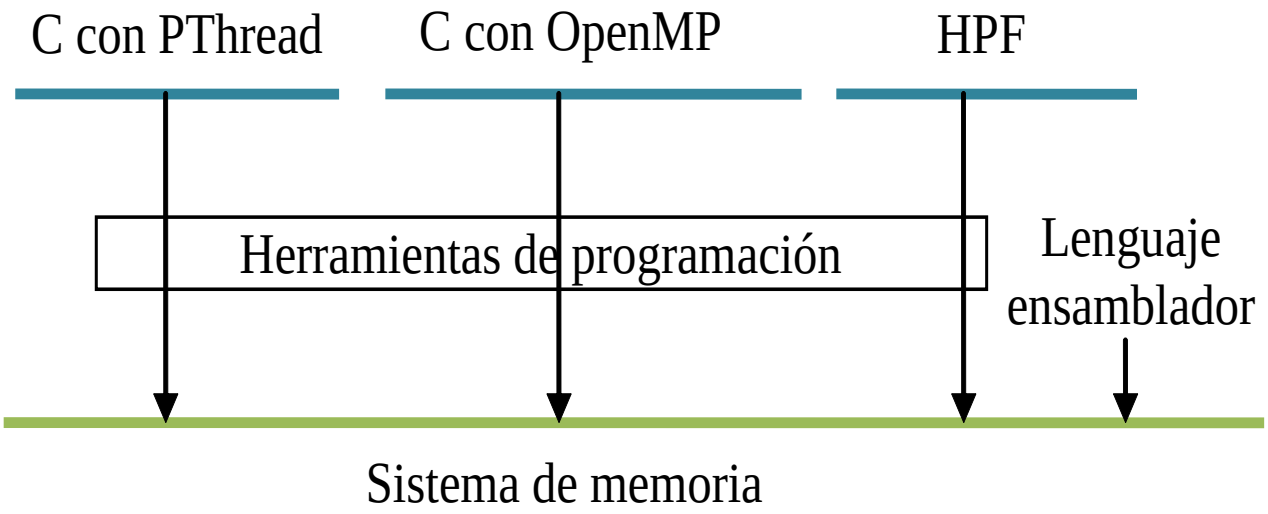
- Fundamental
  - Secc. 10.2. J. Ortega, M. Anguita, A. Prieto. “Arquitectura de Computadores”. ESII/C.1 ORT arq

Contenido Lección 9

- Concepto de consistencia de memoria
- Consistencia secuencial
- Modelos de consistencia relajados

Consistencia de memoria

Modelo de consistencia de memoria Software



Modelo de consistencia de memoria Hardware

- Especifica (las restricciones en) el orden en el cual las operaciones de memoria (lectura, escritura) deben ***parecer*** haberse realizado (operaciones a las mismas o distintas direcciones y emitidas por el mismo o distinto proceso/procesador)
- La coherencia sólo abarca operaciones realizadas por múltiples componentes (proceso/procesador) en una misma dirección

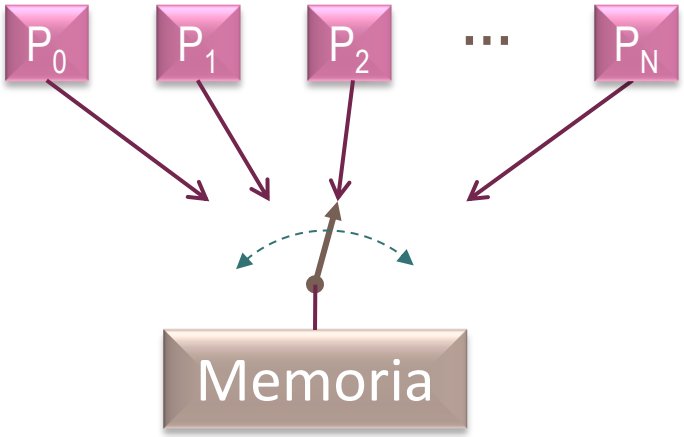
Contenido Lección 9

- Concepto de consistencia de memoria
- Consistencia secuencial
- Modelos de consistencia relajados

Consistencia secuencial (SC)

- SC es el modelo de consistencia que espera el programador de las herramientas de alto nivel
- SC requiere que:
  - Todas las operaciones de un único procesador (thread) parezcan ejecutarse en el orden descrito por el programa de entrada al procesador (orden del programa)
  - Todas las operaciones de memoria parezcan ser ejecutadas una cada vez (ejecución atómica) -> serialización global

Consistencia Secuencial



- SC presenta el sistema de memoria a los programadores como una memoria global conectada a todos los procesadores a través un conmutador central

Consistencia Secuencial

<table>
<tr><td colspan="2">Inicialmente k1=k2=0
<u>P1</u>
k1=1;
if (k2=0) {
  Sección crítica
};</td><td><u>P2</u>
k2=1;
if (k1=0) {
  Sección crítica
};</td><td>1
¿Qué espera el programador?</td></tr>
<tr><td><u>P1</u>
A=1;</td><td>Inicialmente
<u>P2</u>
if (A=1)
  B=1;</td><td>A=B=0

<u>P3</u>
if (B=1)
  reg1=A;</td><td>2
¿Qué espera el programador que se almacene en reg1 si llega a ejecutarse reg1=A?</td></tr>
<tr><td colspan="2">Inicialmente A= 0
<u>P1</u>
A=1;
k=1;</td><td><u>P2</u>
while (k=0) {};
copia=A;</td><td>3
¿Qué espera el programador que se almacene en copia?</td></tr>
</table>

Ejemplo de Consistencia Secuencial

(1) Escribir K1=1

(2) Leer K2 (¿K2==0?)

(a) Escribir K2=1

(b) Leer K1 (¿K1==0?)

1

Orden con Consistencia Secuencial:

(1)(2)(a)(b) (a)(b)(1)(2)

(1)(a)(2)(b) (a)(1)(b)(2)

(1)(a)(b)(2) (a)(1)(2)(b)

(1) Escribir A=1

(2) Escribir K=1

(a) Leer K (while k==0 {})

(b) Leer A

3

Orden con Consistencia Secuencial:

(1)(a)….(a)(2)(a)(b)

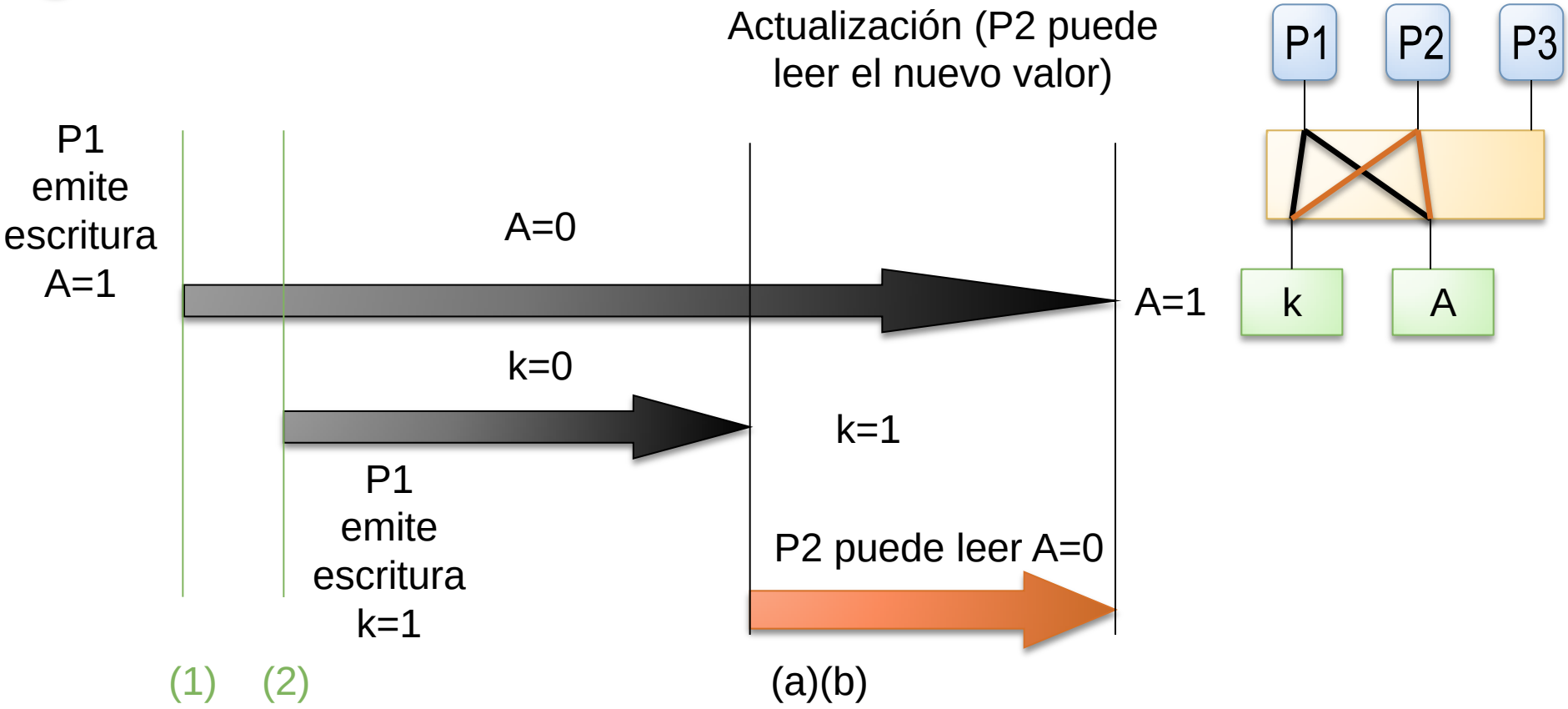
(1)(2)(a)(b)

(a)..(a)(1)(a)..(a)(2)(a)(b)

(a)..(a)(1)(2)(a)(b)

¿Qué puede ocurrir en el computador?

3



No se garantiza el orden W→W

Contenido Lección 9

- Concepto de consistencia de memoria
- Consistencia secuencial
- Modelos de consistencia relajados

Modelos de consistencia relajados

- Difieren en cuanto a los requisitos para garantizar SC que relajan (los relajan para incrementar prestaciones):
  - Orden del programa:
    - Hay modelos que permiten que se relaje en el código ejecutado en un procesador el orden entre dos acceso a distintas direcciones (W→R, W→W, R→RW)
  - Atomicidad:
    - Hay modelos que permiten que un procesador pueda ver el valor escrito por otro antes de que este valor sea visible al resto de los procesadores del sistema
- Los modelos relajados comprenden:
  - Los órdenes de acceso a memoria que no garantiza el sistema de memoria (tanto órdenes de un mismo procesador como atomicidad en las escrituras).
  - Mecanismos que ofrece el hardware para garantizar un orden cuando sea necesario.

Ejemplos de modelos de consistencia hardware relajados

Modelo	Orden relajado W→R	Orden relajado W→W	Orden relajado R→RW	Lec. anticipada de_otro	Lec. anticipada propia	Mecanismos (instrucciones) para garantizar orden global
TSO	Si				Si	lect-modif.-escrit. atómica (l-m-e)
PC	Si			Si	Si	Inst. serialización, l-m-e
PSO	Si	Si			Si	l-m-e, STBAR
WO	Si	Si	Si		Si	sincronización
RCsc	Si	Si	Si		Si	Adquisición, liberación,l-m-e
RCpc	Si	Si	Si	Si	Si	Adquisición, liberación,l-m-e
Alpha	Si	Si	Si		Si	MB, WMB
RMO	Si	Si	Si		Si	MEMBAR
PowerPC	Si	Si	Si	Si	Si	SYNC, ISYNC
Itanium	Si	Si	Si		Si	LD.ACQ, ST.REL, MF, l-m-e

Siguel la tabla de Adve y Gharachorloo en (biblioteca ugr) http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=546611&isnumber=11956

Consistencia secuencial

<table>
<tr><td colspan="2">Inicialmente k1=k2=0
<u>P1</u>
k1=1;
if (k2=0) {
Sección crítica
};</td><td><u>P2</u>
k2=1;
if (k1=0) {
Sección crítica
};</td><td>1
NO se comporta como SC los que relajan el orden W→R</td></tr>
<tr><td><u>P1</u>
A=1;</td><td>Inicialmente
<u>P2</u>
if (A=1)
B=1;</td><td>A=B=0

<u>P3</u>
if (B=1)
reg1=A;</td><td>2
NO se comporta como SC los que no garantizan atomicidad</td></tr>
<tr><td colspan="2">Inicialmente A= 0
<u>P1</u>
A=1;
k=1;</td><td><u>P2</u>
while (k=0) {};
copia=A;</td><td>3
NO se comporta como SC los que relajan el orden W→ W o R → R</td></tr>
</table>

Modelo que relaja W->R

- Permiten que una lectura pueda adelantar a una escritura previa en el orden del programa; pero evita dependencias RAW
  - Lo implementan los sistemas con buffer de escritura para los procesadores (el buffer evita que las escrituras retarden la ejecución del código bloqueando lecturas posteriores)
  - Generalmente permiten que el procesador pueda leer una dirección directamente del buffer (leer antes que otros procesadores una escritura propia)
- Para garantizar un orden correcto se pueden utilizar instrucciones de serialización
- Hay sistemas en los que se permite que un procesador pueda leer la escritura de otro antes que el resto de procesadores (acceso no atómico)
  - Para garantizar acceso atómico se puede utilizar instrucciones de lectura-modificación-escritura atómicas

Modelo que relaja W->R y W->W

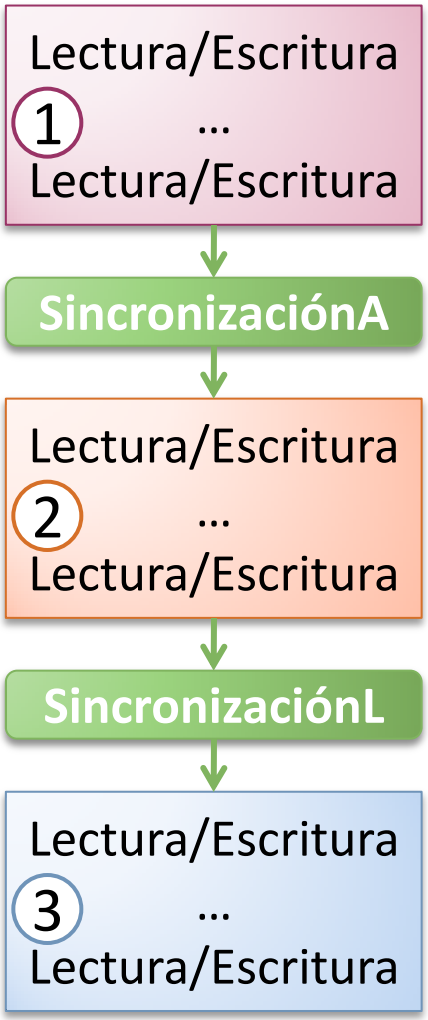
- Tiene buffer de escritura que permite que lecturas adelanten a escrituras en el buffer
- Permiten que el hardware solape escrituras a memoria a distintas direcciones, de forma que pueden llegar a la memoria principal o a caches de todos procesadores fuera del orden del programa.
- En sistemas con este modelo se proporciona hardware para garantizar los dos órdenes. Los sistemas con Sparc implementa un modelo de este tipo.
- Este modelo no se comporta como SC en el siguiente ejemplo:

P1	**P2**
A=1; k=1;	while (k=0) {}; copia=A;

Modelo de ordenación débil

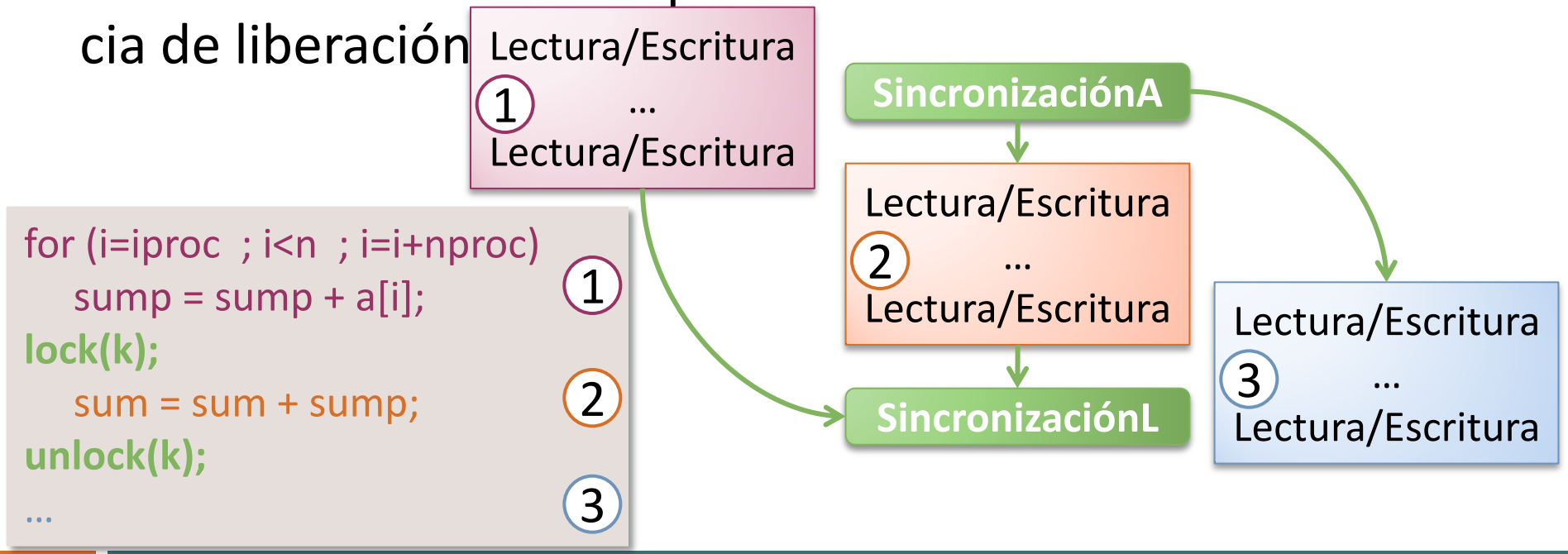
- Relaja W->R, W->W y R->RW
- Si S es una operación de sincronización (liberación o adquisición), ofrece hardware para garantizar el orden:
  - S->WR
  - WR->S
- PowerPC implementa un modelo basado en ordenación débil

```
for (i=iproc  ; i<n  ; i=i+nproc)
    sump = sump + a[i];                    1
lock(k);
    sum = sum + sump;      /* SC */        2
unlock(k);
...                                        3
```



Consistencia de liberación

- Relaja W->R, W->W y R->RW
- Si SA es una operación de adquisición y SL de liberación, ofrece hardware para garantizar el orden:
  - SA->WR y WR->SL
- Sistemas con Itanium implementan un modelo de consistencia de liberación



Para ampliar …

- Artículos en revistas
  - Adve, S.V.; Gharachorloo, K.; , "Shared memory consistency models: a tutorial," *Computer* , vol.29, no.12, pp.66-76, Dec 1996. Disponible en (biblioteca ugr): http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=546611&isnumber=11956