Examen Multiprocesadores

Otoño 2003

Preguntas/Problemas (sin apuntes 60')

Responder razonada y brevemente las siguientes cuestiones:

- Que quiere decir "transferencia cache a cache"

- Que representa el estado Shared en un protocolo MSI. Cuantos procesadores pueden tener una línea en estado M, S , I ?

- Escribir un bucle paralelo OpenMP en el que el protocolo de coherencia MESI se comporte mejor que el MSI.

- Tenemos un sistema NUMA con protocolo MSI basado en directorio. Queremos que en cuanto el directorio recibe una petición de línea en exclusiva (RdX) y contesta con la lista de procesadores que tienen la línea pueda pasar su estado inmediatamente a M. Esto tiene la ventaja de evitar estados intermedios en el directorio y por tanto simplificar su diseño. También nos ahorramos el mensaje de confirmación desde el nuevo propietario de la línea indicando que ya ha realizado todas las transacciones necesarias y que realmente tiene la línea y la esta/ha modificado. Que otras implicaciones tiene?

- Tenemos un sistema de 64 procesadores. Razonar que política (o indicar pros y contras de cada una) escogería entre Gang scheduling (quantum ½ hora, overhead cambio contexto 5') y Backfilling si nuestro objetivo es minimizar el tiempo medio de respuesta y los trabajos (suponemos todos encolados desde el principio) tienen las características siguientes:
  - A: duran 4 horas, piden 64 procesadores cada uno.
  - B: duran 4 horas y piden 32 ó 64 procesadores.
  - C: duran entre 1 y 6 horas y piden 64 procesadores.
  - D: duran entre 1 y 6 horas y piden entre 2 y 64 procesadores

Problema 1

La política que colocación de páginas en un sistema NUMA (128 procesadores, módulos de memoria de 256MB) puede ser "first touch" (en el modulo de memoria del primer procesador que la referencia) o "entrelazado" (paginas consecutivas en módulos de memoria de procesadores consecutivos). Indicar el comportamiento (rendimiento comparativo) de las cuatro combinaciones posibles al ejecutar las dos versiones de código siguientes con cada una de las políticas.

VERSION 1

```
#define N 16*1024*1024
int i,its;
double A[N];

#pragma omp parallel for
for (i=0;i<N;i++) A[i]=0;

for(its=0;its<1000;its++){
#pragma omp parallel for
   for (i=0; i<N; i++)
       A[i]= f(A[i]);
}
```

VERSION 2

```
#define N 16*1024*1024
int i,its;
double A[N];


for (i=0;i<N;i++) A[i]=0;

for(its=0;its<1000;its++){
#pragma omp parallel for
   for (i=0; i<N; i++)
       A[i]= f(A[i]);
}
```

En el caso de "Entrelazado", podría pensare alguna modificación del código que mejorara el rendimiento? Que precauciones habría que tomar?

Problema 2

Analizar el siguiente trozo de programa. Sabemos que ninguno de los valores contenidos en `proc` esta repetido.

```
do i = 1, max
 call mpi_irecv(d,1,MPI_INTEGER,proc(i),i,mpi_comm_world,req,ierr )
 call mpi_send(sum,1,MPI_INTEGER,proc(i),i,mpi_comm_world,ierr )
 call mpi_wait(req,status,ierr )
 sum = sum + d
enddo
```

Preguntas:

- Dar un ejemplo de contenidos de `proc` que cause algún tipo de problema en la ejecución.
- Explicar que se puede lograr en el caso en que max=log P con el contenido adecuado de `proc(i)`. Dar posibles contenidos de los cuatro vectores que logren ese objetivo para el caso P=4, max=2.
- Puede programarse con send y recv?

- Hacer una versión del alltoall de dos vectores A(i), B(i) de P enteros usando isends e irecvs.

Problema 3

Tenemos un diseño de un multiprocesador basado en bus (del tipo de ciclo partido visto en clase) que soporta hasta 16 procesadores conectados a 16 módulos de memoria. Disponemos de dos tipos de procesadores. El procesador A sólo es capaza de lanzar un acceso a memoria. El procesador B puede lanzar varios accesos e incluso admite que se sirvan fuera de orden. Cada módulo de memoria es de 256 Mbytes. El bus soporta un protocolo de coherencia MSI ofreciendo consistencia secuencial. El tamaño de la línea de cache es 128 bytes y la página 4 Kbytes.

Tras analizar el rendimiento nos damos cuenta de que necesitamos más ancho de banda y decidimos hacer un nuevo diseño replicando el bus. Decidimos que cada bus servirá para conectar todos los procesadores con un grupo de 8 módulos de memoria.

Preguntas:

- Tenemos que optar entre mapear las líneas de cache a buses de forma entrelazada (líneas consecutivas en buses alternos) o no entrelazada. Cómo puede influir la decisión en el rendimiento del sistema?
- Que bits de la dirección física hemos de usar para determinar el bus a pedir si se mapea entrelazado? Y si se mapea no entrelazado? Puede el procesador hacer la petición al árbitro del bus en paralelo con la traducción de dirección lógica a física.
- Garantiza el protocolo MSI en cada bus la coherencia en el sistema completo si se usa el procesador A? y si se usa el B?
- Garantiza el protocolo MSI en cada bus la consistencia secuencial en el sistema completo si se usa el procesador A? y si se usa el B?
- Comentar las modificaciones que habría que hacer en el controlador de "snoop" y cache? (no es necesario llegar al último detalle pero sí identificar la problemática i razonar sobre opciones de diseño)

Pensamos una segunda opción en la que los dos buses permiten conectar cualquier procesador a cualquier módulo de memoria. Cada procesador decide **aleatoriamente** para cada acceso qué bus utilizar. Razonar sobre los siguientes puntos:

- Que posibles ventajas o inconvenientes puede tener sobre la primera opción en cuanto a rendimiento? Se podría hacer alguna mejora en esta opción que aumentara en rendimiento?
- Comentar las problemática de diseño que genera la opción 2