Examen Multiprocesadores

Primavera 2004

Problema 1

Paralelizar con MPI el siguiente programa para un número de procesadores fijo (N_PROC. Constante que podemos utilizar en el código). Suponer que N es múltiplo de N_PROC.

```
program LU
integer i, j, k
double precision A(N, N)

do k = 1, N
  do i = k + 1, N
    A(i, k) = A(i, k) / A(k, k)
  enddo
  do j = k + 1, N
    do i = k + 1, N
      A(i, j) = A(i, j) - A(i, k) * A(k, j)
    enddo
  enddo
enddo
end
```

1) Declarar las estructuras de datos, explicando posibles distribuciones de datos y cuales son las ventajas o inconvenientes en caso de haber distintas alternativas.
2) Realizar una primera versión del código sin utilizar llamadas MPI no bloqueantes.
3) Se puede obtener alguna ventaja utilizando llamadas no bloqueantes?

Problema 2

Disponemos de un multiprocesador de memoria compartida que corre Unix/linux. Queremos implementar la librería de MPI aprovechando el soporte hardware de nuestra máquina. Explicar para cada una de las llamadas siguientes la estructura del código que las implementa, indicando en su caso las llamadas a sistema operativo (u otras rutinas de librería que normalmente ofrecen los sistemas de máquinas con memoria compartida) que se deban utilizar. Describir las estructuras de datos principales de vuestra librería MPI y donde residen.

1) MPI_comm_rank.
2) MPI_initialize.
3) MPI_send, MPI_recv

4) Argumentar cuantos fallos de cache e invalidaciones se producirán cuando un proceso envía un mensaje de 128000 bytes a otro en un sistema con líneas de 128 bytes? Qué secuencias de transiciones de estado se producirán para cada línea?
5) Qué técnicas podrían usarse en la implementación del send/recv para aumentar la velocidad de la comunicación para mensajes grandes?
6) Que soporte podría dar el sistema operativo para reducir el número de copias memoria-memoria?

Problema 3

Tenemos la traza de dos versiones de un programa. Una versión es MPI y la otra OpenMP.

Versión OpenMP:

1) a partir de las vistas siguientes (general y zoom de la zona intermendia) escribir la estructura del código fuente. (Usar los nombres rutina_blanca, rutina_gris rutina_no_instrumentada)

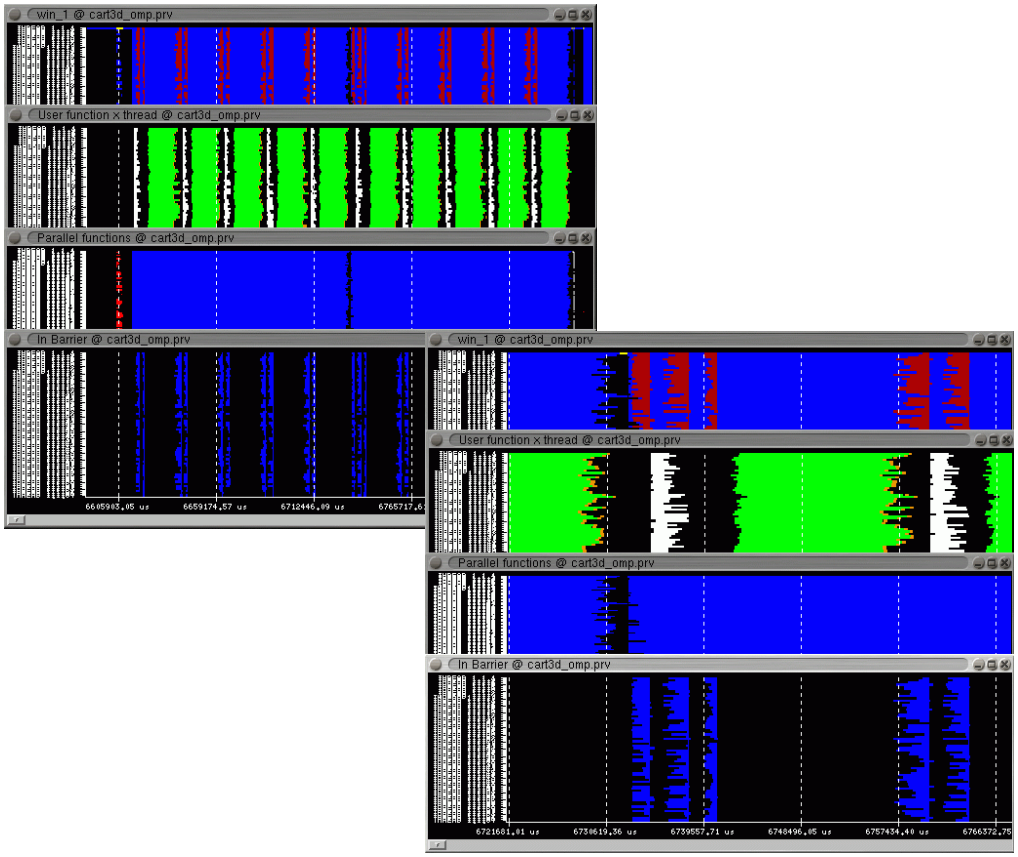
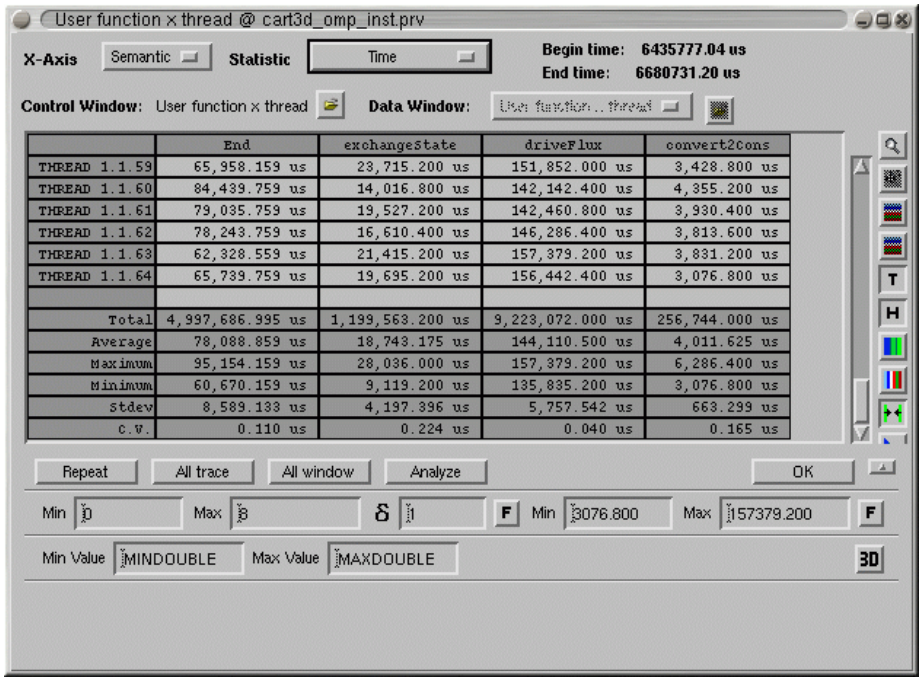


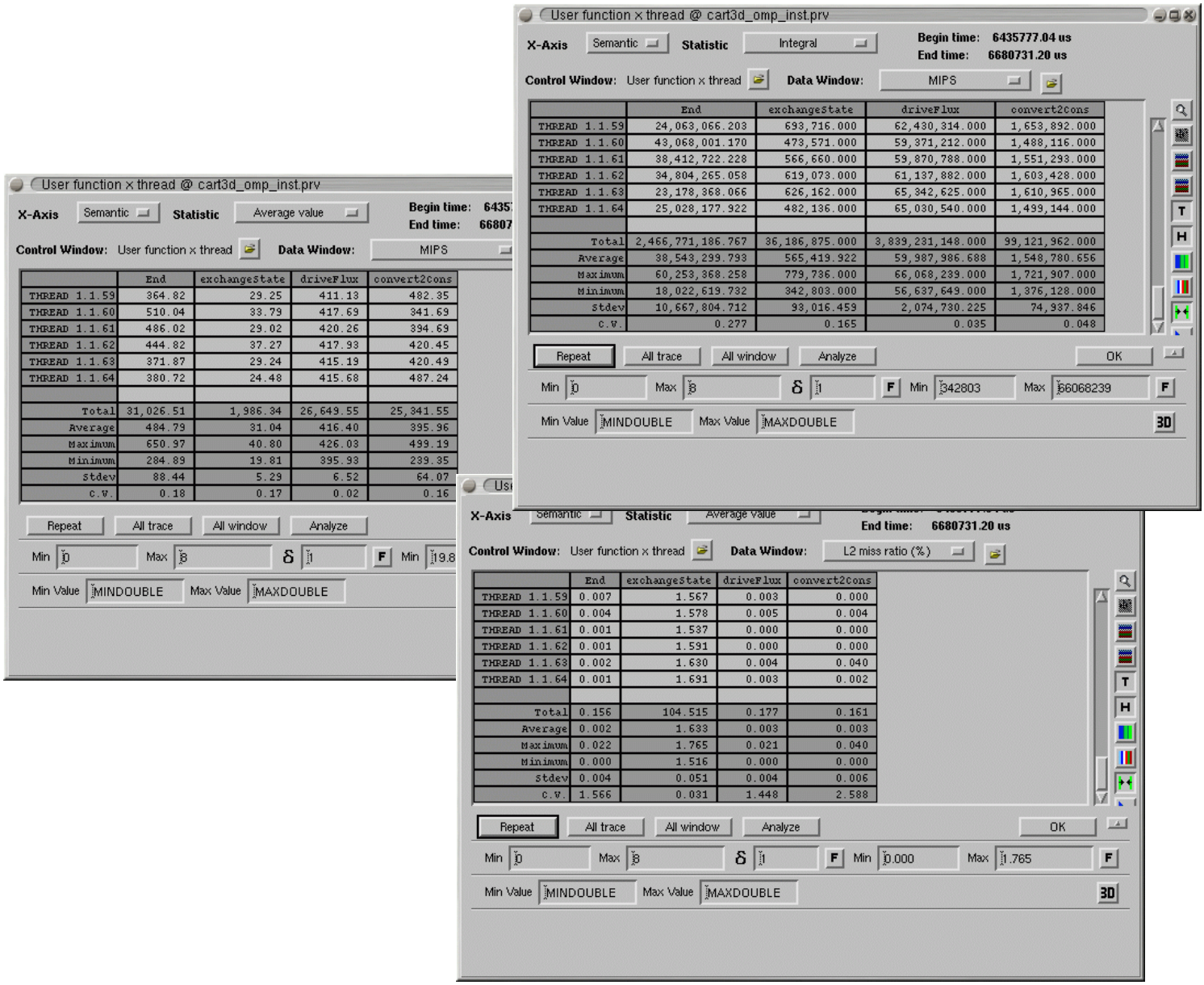
Figure 1

2) A la vista de la tabla siguiente decir el nombre de la rutina blanca y gris (y por que).



	End	exchangeState	driveFlux	convert2Cons
THREAD 1.1.59	65,958.159 us	23,715.200 us	151,852.000 us	3,428.800 us
THREAD 1.1.60	84,439.759 us	14,016.800 us	142,142.400 us	4,355.200 us
THREAD 1.1.61	79,035.759 us	19,527.200 us	142,460.800 us	3,930.400 us
THREAD 1.1.62	78,243.759 us	16,610.400 us	146,286.400 us	3,813.600 us
THREAD 1.1.63	62,328.559 us	21,415.200 us	157,379.200 us	3,831.200 us
THREAD 1.1.64	65,739.759 us	19,695.200 us	156,442.400 us	3,076.800 us
Total	4,997,686.995 us	1,199,563.200 us	9,223,072.000 us	256,744.000 us
Average	78,088.859 us	18,743.175 us	144,110.500 us	4,011.625 us
Maximum	95,154.159 us	28,036.000 us	157,379.200 us	6,286.400 us
Minimum	60,670.159 us	9,119.200 us	135,835.200 us	3,076.800 us
Stdev	8,589.133 us	4,197.396 us	5,757.542 us	663.299 us
C.V.	0.110 us	0.224 us	0.040 us	0.165 us

3) Comentar (balanceo de carga, rendimiento,…) las tablas siguientes.

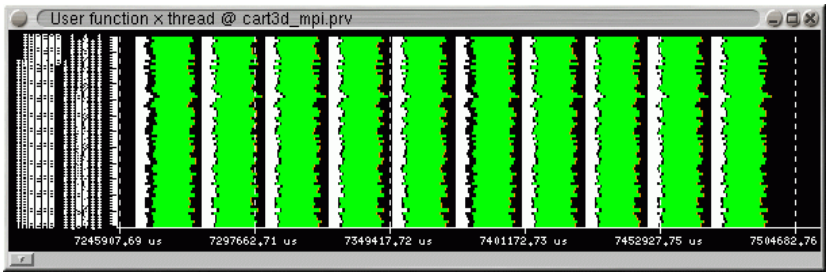


4) Sabiendo que la rutina blanca realiza esencialmente movimientos de datos entre distintos arrays, que en promedio cada procesador viene a mover unos 750Kbytes en total y que el tamaño de la linea de cache es 128 bytes razonar si es posible que haya false sharing o no.

5) Estimar el coste medio de un fallo de cache. Sabiendo que el nominal de un acceso local es de unos 300ns y el de un acceso remoto puede ser unas 3 veces superior, comentar el valor obtenido.

Versión MPI

6) La siguiente vista es la correspondiente a la segunda de la figura XX. Comentar las diferencias y explicar



La siguiente figura muestra las llamadas MPI en el periodo equivalente, el zoom a una sola iteración (mostrando solo los mensajes que salen de dos de los procesadores) y a unos cuantos threads.

7) Escribir la estructura del código.
8) Escribir una versión alternativa usando llamadas no bloqueantes.