Entregable - CPA

López Muñoz, Alejandro

Benites Aldaz, Dairon Andrés

daibeal@inf.upv.es - allomuo@inf.upv.es

Índice

daibeal@inf.upv.es - allomuo@inf.upv.es

Ejercicio 1

Obtén una primera versión paralela del programa paralelizando las funciones que utiliza la función process: *distance* y *swap*. Llama a esta versión restore1.c

Versión secuencial

```
// Compute the difference between two (horizontal or vertical) lines of an image
// a1 and a2 are the two lines. Each of them is n pixels long
int distance( int n, Byte a1[], Byte a2[], int stride ) {
  int d,i,j, r,g,b;
  stride *= 3;
  d = 0;
  for ( i = 0 ; i < n ; i++ ) {
    j = i * stride;
    r = (int)a1[j]   - a2[j];   if ( r < 0 ) r = -r;  // Difference in red
    g = (int)a1[j+1] - a2[j+1]; if ( g < 0 ) g = -g;  // Difference in green
    b = (int)a1[j+2] - a2[j+2]; if ( b < 0 ) b = -b;  // Difference in blue
    d += r + g + b;
  }
  return d;
}
```

```
// Swap two rectangles (a1 and a2) of an image.
// rw and rh define the width and height of both rectangles
// w is the width of the complete image that contains the rectangles
void swap( Byte a1[],Byte a2[],int rw,int rh,int w ) {
  int x,y,d;
  Byte aux;
  if ( a1 != a2 ) {
    rw *= 3; w *= 3; // Each pixel is 3 bytes
    for ( y = 0 ; y < rh ; y++ ) {
      // Swap row y of the two rectangles
      d = w * y;
      for ( x = 0 ; x < rw ; x++ ) {
        // Swap a single byte of the two rows
        aux = a1[d+x];
        a1[d+x] = a2[d+x];
        a2[d+x] = aux;
      }
    }
  }
}
```


daibeal@inf.upv.es - allomuo@inf.upv.es

Versión paralela

daibeal@inf.upv.es - allomuo@inf.upv.es

Ejercicio 2

Obten una segunda versión paralela paralelizando el cuerpo de la función process, dejando la función swap paralelizada (pero no la función distance). Observa que los bucles externos de la función process no se pueden paralelizar, con lo que deberás paralelizar los bucles internos. Llama a esta versión restore2.c.

daibeal@inf.upv.es - allomuo@inf.upv.es

Ejercicio 3

En este ejercicio no se pide que modifiques ningún código ni que realices ninguna ejecución, sino que respondas de forma razonada a lo que se pregunta.

En la paralelización del bucle y2 de process, ¿crees que con alguna planificación se obtendría un mejor equilibrio de carga que con otra? ¿Sí/no? ¿Cuáles? ¿Por qué?

¿Y en el bucle y de process? (como se ha dicho, ese bucle no se puede paralelizar, pero para este ejercicio teórico supongamos que sí se puede).

daibeal@inf.upv.es - allomuo@inf.upv.es

Ejercicio 4

Utilizando los nodos de cálculo del cluster kahan, saca tiempos de ejecución de las dos versiones paralelas realizadas, usando 16 hilos y las siguientes planificaciones:

- **static con tamaño de *chunk* por defecto**

- **static con tamaño de *chunk* 1.**

- **dynamic con tamaño de chunk por defecto**

Analiza cuál es la mejor planificación en cada versión paralela, indicando a qué puede deberse.

daibeal@inf.upv.es - allomuo@inf.upv.es

Ejercicio 5

Utilizando los nodos de cálculo del cluster kahan, saca tiempos de ejecución de las dos versiones paralelas realizadas, variando el número de hilos y eligiendo en cada versión la planificación con la que se hayan obtenido mejores resultados en el ejercicio anterior. Para limitar el número de ejecuciones, se recomienda usar potencias de 2 para los valores del número de hilos (2, 4, 8...), llegando hasta el número de hilos que consideres adecuado (justifica por qué eliges ese número máximo de hilos).

daibeal@inf.upv.es - allomuo@inf.upv.es

Ejercicio 6

En este ejercicio hay que hacer que cada hilo muestre información sobre las iteraciones que le ha tocado procesar de un bucle paralelizado. En concreto, partimos de la versión paralela del ejercicio 2, donde, en cada iteración del bucle y, se reparten las iteraciones del bucle y2 entre los hilos. En principio habría que hacer que, en cada iteración del bucle y, cada hilo muestre un mensaje con su identificador, cuántas iteraciones ha procesado del bucle y2 (en esa iteración del bucle y) y cuáles han sido la menor y mayor distancias que ha encontrado en esas iteraciones. Sin embargo, para evitar que salgan demasiados mensajes por pantalla, haz que solo se muestren los mensajes correspondientes a la primera iteración del bucle y.

daibeal@inf.upv.es - allomuo@inf.upv.es

Anexos

daibeal@inf.upv.es - allomuo@inf.upv.es