ASIGNATURA:
Computación de Altas Prestaciones

Paralelización de código con OpenMP+MPI

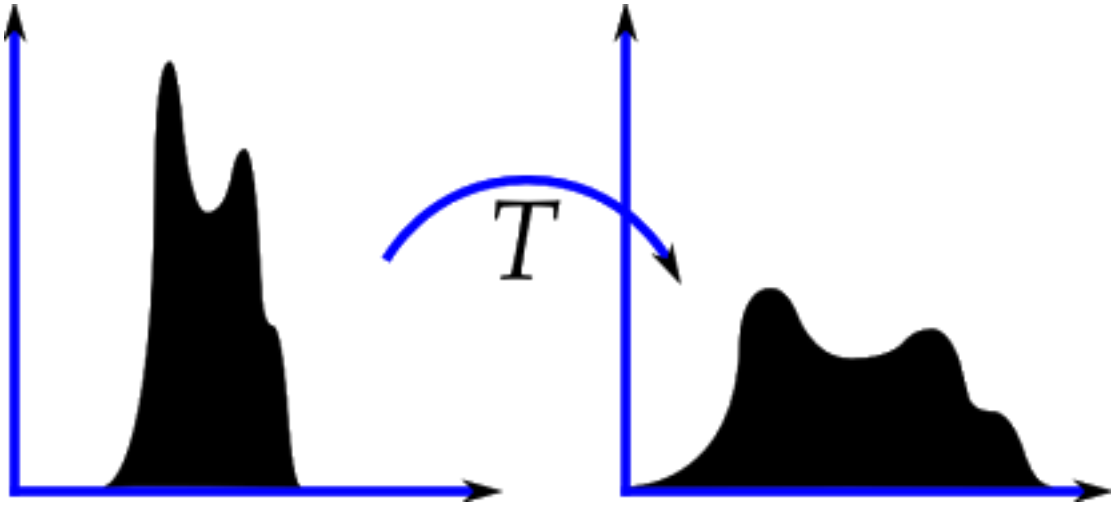
La mejora de contraste es una operación común en el procesamiento de imagen. Es un método útil para el procesamiento de imágenes científicas, tales como imágenes de rayos X o imágenes obtenidas por satélite. También es una técnica útil para mejorar los detalles en las fotografías que están sobre o subexpuestas.

El objetivo de este trabajo es desarrollar una aplicación de mejora de contraste que utiliza la aceleración híbrida sobre OpenMP y MPI.

En este documento, se dan las ideas básicas y los principios de mejora de contraste mediante modificación del histograma. Empezamos con el caso simple, realce de contraste para las imágenes en escala de grises. Entonces, tratamos de aplicar el método similar a las imágenes en color.

Una implementación sencilla C se da como referencia y punto de partida.

El histograma de una imagen digital representa su distribución tonal. La ecualización del histograma indica los valores de intensidad más frecuentes. Esto permite conocer las áreas de menor contraste para obtener un mayor contraste sin afectar el contraste global de la imagen. La siguiente imagen muestra el efecto de la ecualización del histograma en el histograma de la imagen.



A continuación, se muestra un ejemplo de ecualización del histograma, la imagen de la izquierda es la imagen original y la de la derecha es el resultado después de aplicar la ecualización del histograma.

Paralelización de código con OpenMP+MPI
MASTER EN INGENIERÍA INFORMÁTICA

Ecualización de histograma se puede realizar en los siguientes pasos:

- Calcular el histograma de la imagen de entrada;
- Calcular la distribución acumulativa del histograma;
- Utilización de la distribución acumulativa para construir una tabla de búsqueda que mapea cada valor de gris a la ecualizada (este paso se puede combinar con la última);
- Actualización de la imagen utilizando la tabla de búsqueda construida en el último paso.

Material de apoyo

Se proporciona como punto de partida y referencia una implementación sencilla en C.

Requisitos de programación

Se debe implementar un programa paralelo en OpenMP+MPI que realice eficientemente los cálculos. Se proporciona una versión secuencial del programa.

Algunas observaciones sobre la ejecución de la aplicación:

- La aplicación utiliza *pgm* (para imágenes en escala de grises) y *ppm* (para imágenes en color). Por defecto, el nombre de archivo del archivo de entrada es in.pgm / in.ppm.
- Dado que el tamaño de las imágenes ppm/pgm es demasiado grande, se dispone de la siguiente imagen en formato JPG. Una vez descargada la figura será necesario convertirla al formato ppm/pgm (se puede utilizar el comando convert).

El programa se ejecuta de la siguiente manera:

```
mpirun -np X ./contrast
```

Para signara nodos y ejecutarlo en distintos nodos, siga las instrucciones del manual que se ha proporcionado, dado que es necesario asignar antes los nodos.
Si se quiere usar varion nodos hay dos opciones:

1.- Conseguir primero los nodos y luego ejecutar el programa con mpirun. Se puede asignar N nodos y crear Y procesos.

```
salloc -N X
mpirun -np Y./contrast
```

2.- Ejecutarlo directamente con "`srun`" indicando los nodos que se desea. Esto llama a `salloc` y `mpirun` de forma inmediata para asignar N nodos y crear Y procesos.

```
srun. -N X -n Y./contrast
```

La **versión paralela del programa consistirá en los siguientes pasos**:

- Los procesos deben computar una sección reducida para mejorar la carga de cómputo de cada proceso.
- Únicamente un proceso (rank 0 por ejemplo) leerá las imágenes y escribirá los resultados en el disco. Además, un único proceso presentará el tiempo total de la aplicación. ▪ Se valorará positivamente presentar el código comentado.

Para la fase de experimentación, se contará con el sistema *avignon* (avignon.lab.inf.uc3m.es). Este clúster docente consta de 6 nodos de cómputo y cada uno de ellos dispone de cuatro núcleos físicos. En total se dispone de 24 núcleos.

Requisitos de entrega

Programa paralelo. Código fuente.

Se deben tomar medidas para 1, 2, 4 y 6 nodos y estudiar el efecto de la paralelización sobre el código inicial.
La implementación realizada consistirá en la paralelización empleando los entornos MPI y OpenMP.

- Se deben hacer y entregar dos versiones del programa:
  1.- Versión con MPI pura, sin OpenMP. Nombre archivo: "contrast-mpi.cpp"
  2.- Versión con MPI + OpenMP. Nombre archivo: "contrast-mpi-openmp.cpp"

Se debe elaborar una **memoria** incluyendo:

- Una descripción de la metodología de paralelización basada en cuatro fases: descomposición, asignación, orquestación y reparto/mapeo.
- Gráficas de las medidas de tiempo y aceleración de cada versión y comparativa de las mismas.
- Interpretación de las gráficas.
- Conclusiones sobre la fase de experimentación realizada.

Se debe incluir un. Fichero "**autores**" con una línea por autor con formato:
APELLIDOS, NOMBRE, NIA

Recomendaciones

- Usar el comando "**convert**" para poder generar y visualizar las imágenes procesadas. Por ejemplo

```
convert highres.jpg  in.pgm
```

- Para tomar tiempos se recomienda usar la función `MPI_Wtime` () de la siguiente forma:

```
double tstart = MPI_Wtime();
        …
      código
      …
 double tfinish = MPI_Wtime();
double TotalTime = tfinish - tstart;
```

- Para asegurarse del correcto funcionamiento de la versión paralela, en comparación con la versión secuencial, seguid los siguientes pasos:

1. Ejecutar la versión secuencial con un conjunto de datos de entrada.
2. Ejecutar las diferentes versiones paralelas con el mismo conjunto de datos de entrada que la versión secuencial.
3. Comparar los conjuntos de datos de salida de cada versión paralela con el obtenido en la versión secuencial usando el comando *diff* (man diff).
4. Si el comando *diff* detecta diferencias entre el fichero de salida de la versión secuencial y alguno de los ficheros de salida de las versiones paralelas se considerará que el programa no funciona correctamente y, por tanto, habrá que revisar el algoritmo paralelo.

Compilación

```
$ tar zxvf CAP2022.tgz
$ cd CAP2022
$ make
```

Fecha de entrega

18 de diciembre de 2022 23:59

Material a entregar: archivo comprimido nombre. **CAP-practica2.zip** incluyendo:

- Código paralelo. Un directorio incluyendo las dos versiones y el archivo cMake modificado con todo lo necesario para que compilen usando make.
- Memoria
- Fichero autores