Tarea 4: Introducción a la computación de alto rendimiento (7%)

Marlon Brenes*

El propósito de esta tarea es practicar los conceptos de paralelismo de memoria compartida y distribuida. La tarea involucra entregar código fuente de C++ (utilizando bibliotecas especializadas) con su respectivo reporte detallando el análisis. La tarea se puede realizar en parejas, pero **cada estudiante debe entregar sus propios archivos. El reporte se confecciona por cada estudiante de forma individual**. Usted debe entregar los siguientes archivos para ser evaluados:

- `mandelbrot.cpp`
- `ping_pong.cpp`
- reporte4.pdf/reporte4.doc/reporte4.docx/reporte4.ipynb (dependiendo de cual formato escoja para su reporte)

PARTE I: MEMORIA COMPARTIDA (4.0%)

El conjunto de Mandelbrot es un conjunto de dos dimensiones que exhibe comportamiento caótico y fractal. A pesar de la sencillez de la regla que genera el conjunto, la complejidad asociada al conjunto es altamente magnificada. El conjunto está definido en el plano complejo como el conjunto de números c tales que la función

$$f_c(z) = z^2 + c \tag{1}$$

no diverge al infinito cuando se inicia la iteración con $z = 0$. Esto es, cuando la secuencia $[f_c(0), f_c(f_c(0)), \cdots]$ permanece acotada en su valor absoluto.

Al visualizar los números c que satisfacen esta condición en el plano complejo (Fig. 1), se obtiene un **fractal**. Note que $\mathrm{Re}[c] \in [-2, 1]$ y $\mathrm{Im}[c] = [-1, 1]$.

- Se le ha proporcionado una aplicación en `C++` que genera el conjunto de Mandelbrot e imprime los resultados en formato `ASCII` en la terminal
- La aplicación es muy rudimentaria, dado que no se imprimen distintos símbolos dependiendo del valor de acotación mencionado anteriormente. Lo que hace es imprimir '#' o '.' dependiendo si c da lugar a iteraciones acotadas después de cierto valor de corte. Para visualizar el conjunto en la terminal, asegúrese de que la terminal tenga el tamaño de al menos 155x50 caracteres

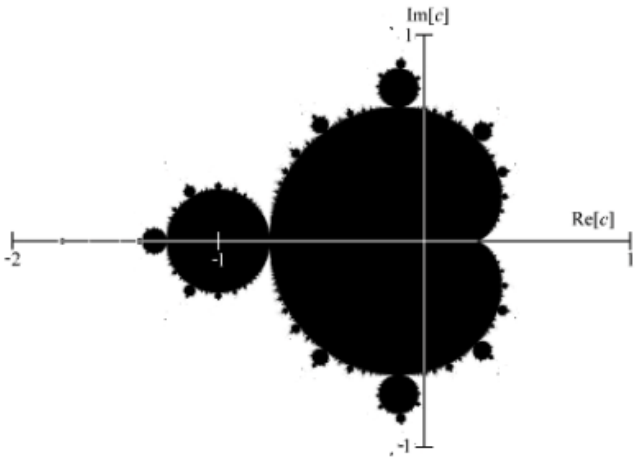

FIG. 1. Conjunto de Mandelbrot

* marlon.brenes@utoronto.ca

- No obstante, la aplicación es suficiente para observar los detalles mínimos del set de Mandelbrot
- **Su tarea** es encontrar una metodología para paralelizar el procedimiento que genera el conjunto utilizando OpenMP en el paradigma de memoria compartida y medir su aceleración (gráfico de speed-up) utilizando hasta 8 procesadores [4.5%]
- Sus tareas son las siguientes:
  - Reproducir el conjunto de Mandelbrot mediante símbolos en la terminal utilizando la aplicación que se le ha entregado y entender la funcionalidad
  - Paralelizar el algoritmo con base en una subdivisión de las **filas** del diagrama (i.e., subdividiendo la región de trabajo en hilos sobre el eye **complejo** del conjunto
  - Paralelizar el algoritmo con base en una subdivisión de las **columnas** del diagrama (i.e., subdividiendo la región de trabajo en hilos sobre el eye **real** del conjunto
  - Una aplicación en paralelo debe dar lugar a los mismos resultados que una aplicación en serie: su aplicación acelerada con OpenMP **debe generar el mismo gráfico en la terminal al que genera la aplicación en serie**
  - Una vez que ha verificado que la aceleración da lugar a resultados correctos, es momento de escalar el algoritmo a grillas más grandes
  - Elimine (comente) la sección del código que imprime el resultado en terminal
  - Imagine que ahora queremos dibujar el conjunto en con una resolución 4K, i.e., 3840 x 2160 pixeles (ancho x alto)
  - Realice los gráficos de escalabilidad utilizando hasta 8 hilos para cada una de las implementaciones (subdividiendo columnas y filas)
  - Explique sus resultados en su reporte

La escalabilidad depende de la arquitectura computacional. Se recomienda utilizar las máquinas del laboratorio de cómputo para evaluar la escalabilidad.

PARTE II: MEMORIA DISTRIBUIDA (3.0%)

Su tarea es implementar la operación de BLAS llamada `AXPY` utilizando paralelismo de memoria distribuida. La operación corresponde a

$$\mathbf{y} := \mathbf{y} + \alpha\mathbf{x}, \tag{2}$$

donde $\mathbf{y}$ y $\mathbf{x}$ son vectores columna y α es un escalar. Utilice números reales de doble precisión. Sus tareas son las siguientes:

- Realice la subdivisión de la región de trabajo por proceso utilizando el algoritmo de `nlocal` y `rest` que vimos en clase
- Cada proceso contiene **solamente los elementos del vector que le corresponden** en memoria
- Inicialice los vectores $\mathbf{y}$ y $\mathbf{x}$ con los valores que desee
- Realice la operación algebraica
- Naturalmente, el resultado final está dividido entre cada uno de los procesos. No es necesario acumular el resultado final en un proceso líder
- Verifique que el resultado es correcto
- Una vez que ha verificado que el resultado es correcto, elimine (comente) la sección del código que imprime los vectores
- Realice la operación utilizando vectores de tamaño 5×10^8 elementos. Cuanta memoria requiere cada proceso para realizar la operación?
- Realice un gráfico de escalabilidad utilizando hasta 8 procesos

La escalabilidad depende de la arquitectura computacional. Se recomienda utilizar las máquinas del laboratorio de cómputo para evaluar la escalabilidad.