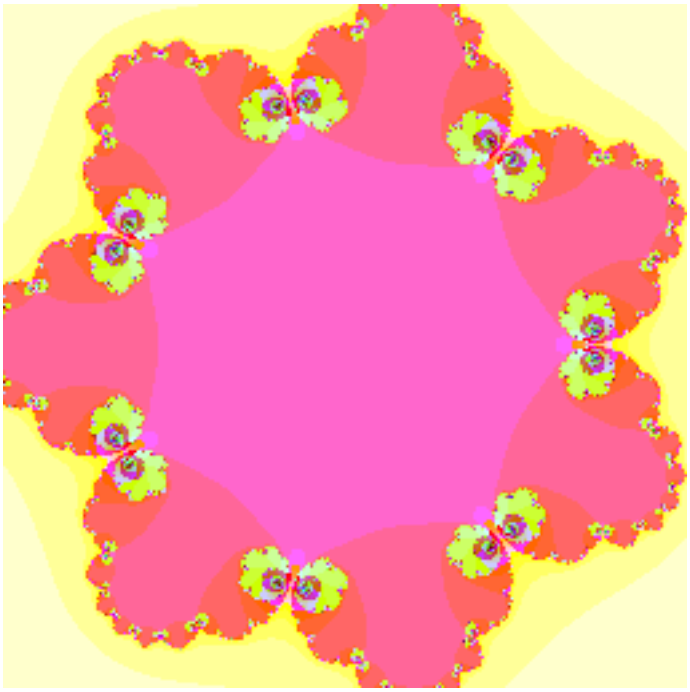
PRÁCTICA 7: Comparativa

Sofía Fernández Moreno

Curso 2016/2017
ACAP

Se va a realizar la comparación de tres métodos para la resolución de un problema para generar fractales, concretamente Conjuntos de Julia para $f(z)=z^7 + c$.

Estos métodos son: el modelo SISD (secuencial), el modelo MIMD y memoria distribuída(MPI) y memoria compartida y SIMD (CUDA).
Los tamaños de imagen a calcular han sido 256, 512, 1024 y 4096. Más adelante veremos que para ciertos tamaños no es posible calcular su imagen. He realizado tres pruebas para cada método y tamaño, con esto obtendremos además la ganancia para MPI y CUDA.
Todas las pruebas se han realizado en los PCs del aula 2.9.
El fractal generado para f(z) es el siguiente:



1. Secuencial

Se ha utilizado sólo un núcleo, por lo que no hay paralelismo. Tras la experimentación ha demostrado ser útil para tamaños pequeños y es la más sencilla de implementar.

Tamaño Ventana	CPU (ms)
256	860
256	860
256	860
512	3450
512	3450
512	3460
1024	13810
1024	13880
1024	13790
2048	55030
2048	55360
2048	55530
4096	Violacion de segmento
4096	Violacion de segmento
4096	Violacion de segmento

2. MPI

Los tiempos que se han tomado con este modelo se han tomado con trece procesos, es decir, con trece procesadores. Los tiempos son bastante altos, principalmente por el tiempo de creación y sincronización de procesos, además de la comunicación final de resultados. Todo esto lleva a que para tamaños a partir de 2048, no pueda calcular la imagen.

Tamaño Ventana	MPI (ms) Para 13 procesos
256	556,3280582
256	323,7190247
256	341,8180943
512	1434,832096
512	1331,619024
512	1197,464228
1024	5058,46405
1024	4894,608021
1024	5108,263016
2048	No puede calcular
2048	No puede calcular
2048	No puede calcular
4096	No puede calcular
4096	No puede calcular
4096	No puede calcular

3. CUDA

Para esta implementación se ha utilizado la GeForce 210.
Se crea una hebra por cada pixel de la imagen, por lo que se crea un bloque por cada fila de pixeles de la imagen. Tuve problemas en cuanto a la implementación rutinaria, de crear hebras por bloques como columnas tiene la imagen, ya que para un tamaño superior a 512 no podía calcular el fractal. Por tanto, he establecido un tamaño de bloque fijo, este será de 512 hebras, para cuando tengamos más de 512 columnas, se creará los bloques necesarios para que pueda calcular la imagen.

Tamaño Ventana	CUDA (ms)
256	16,596001
256	15,264
256	16,427
512	86,774002
512	85,390999
512	85,559998
1024	289,903992
1024	294,820007
1024	293,621002
2048	943,737
2048	938,244019
2048	940,200012
4096	3292,970947
4096	3297,387939
4096	3298,731934

4. Comparación con ganancia

Primero realizamos las medias con las pruebas obtenidas anteriormente, con los tiempos medimos pasamos a calcular la ganancia

Tamaño Ventana	CPU (ms)	MPI (ms) Para 13 procesos	CUDA (ms)
256	860	407,2883924	16,095667
512	3453,333333	1321,305116	85,908333
1024	13826,66667	5020,445029	292,781667
2048	55306,66667	No puede calcular	940,7270103
4096	Violacion de segmento	No puede calcular	3296,363607

Ganancia

Tamaño Ventana	MPI (ms) Para 13 procesos	CUDA (ms)
256	2,111525926	53,43052885
512	2,613577509	40,1978855
1024	2,754071917	47,22517912
2048	No puede calcular	58,79140926
4096	No puede calcular	No puede calcular

En conclusión, la mejor opción que se nos plantea sería la versión realizada en CUDA.

Para la versión secuencial, se puede apreciar en los resultados, existe un tamaño de ventana desde el cual el algoritmo secuencial empieza a despuntar de forma exponencial debido a limitaciones hardware.

En cambio, esto no sucede con el algoritmo CUDA, el cual
se mantiene en unos tiempos bastante bajos con respecto a los demás, ya que este va creciendo muy lentamente.

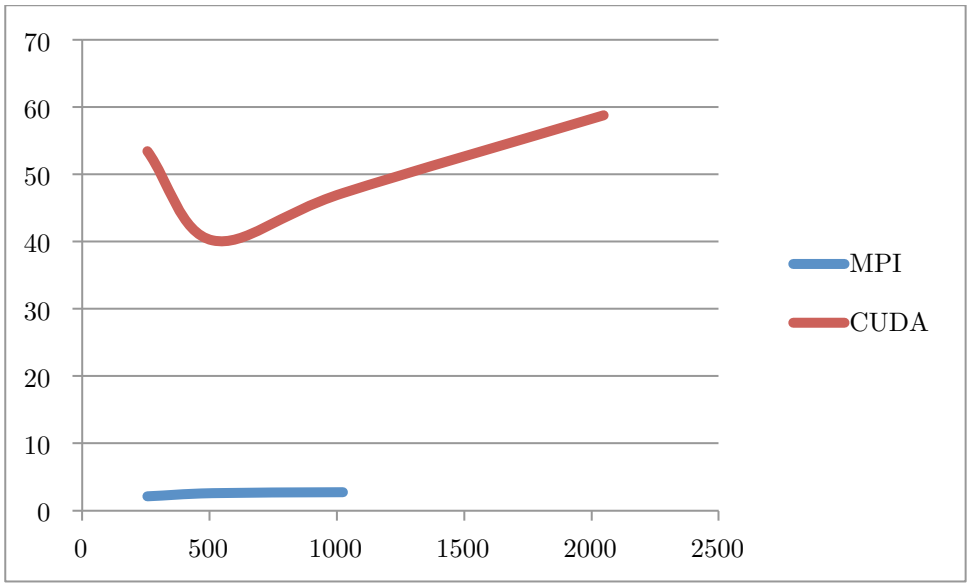


Ilustración 1 Ganancia de CUDA y MPI