## Indice



### ***Bibliografía*** *y enlaces para el aprendizaje autónomo*

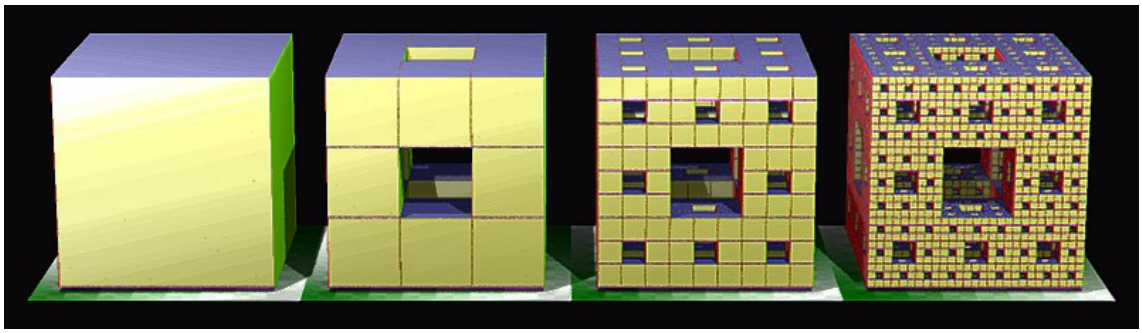
| | |
|---|---|
| Página web OpenMPI | **http://www.open-mpi.org** |
| Página web MPI | **http://www.mpi-forum.org/** |
| Página web gnuplot | **http://www.gnuplot.info** |
| Fractal en Wikipedia | **http://es.wikipedia.org/wiki/Fractal** |

"Parallel Programming. Technique and Applications Using Networked Workstations and Parallel Computers" de B. Wilkinson y M. Allen.

## *Descripción de la práctica*

El matemático francés Benoit Mandelbrot acuñó la palabra **fractal** en la década de 1970, derivándola del adjetivo latín fractus. El correspondiente verbo latino: frangere, significa romper, crear fragmentos irregulares e impulsó un cambio decisivo en el estudio de los fractales con el descubrimiento de la geometría fractal.
Los fractales poseen las propiedades de autosemejanza y autorreferencia. La autosemenjanza se refiere al hecho de que un elemento fractal posee la misma apariencia que un segmento del mismo elemento, es decir, que la apariencia del elemento fractal no depende del grado de ampliación que tomemos del mismo. La autorreferencia nos indica la manera recursiva de generar fractales en la que el propio elemento fractal está incluido en la definición de si mismo. Con estas dos propiedades podemos asegurar la capacidad de generar una ampliación de un elemento fractal infinitas veces. De este modo siempre se podrá extraer una porción de un elemento fractal y generar su ampliación.

Un ejemplo es la **esponja de Menger**, el cual es un conjunto fractal descrito por primera vez en 1926 por Karl Menger mientras exploraba el concepto de dimensión topológica. Este conjunto constituye una generalización del conjunto del cantor, siendo esta una generalización tridimensional:



La construcción de la esponja de Menger se define de forma recursiva, comenzando con un cubo completo (figura 1), a continuación se divide cada cara del cubo en 9 cuadrados, subdividiendo el cubo en 27 cubos más pequeños. Luego se eliminan los 6

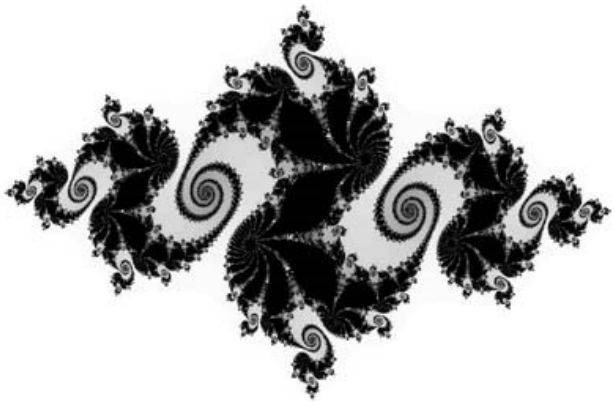
cubos centrales de cada cara y el cubo central dejando solamente 20 cubos (figura 2). Finalmente repetimos estos pasos para cada uno de los 20 cubos restantes (figura 3 y 4). La esponja de Menger es el límite de este proceso tras un número infinito de iteraciones.

Otro ejemplo de fractales son los Conjuntos de Julia, llamados así por el matemático francés Gastón Julia, resultan de evaluar el comportamiento de funciones holomórficas en el espacio de números complejos. El caso más sencillo es el del polinomio:

$$\mathbf{f(z)= z^2 +c}$$

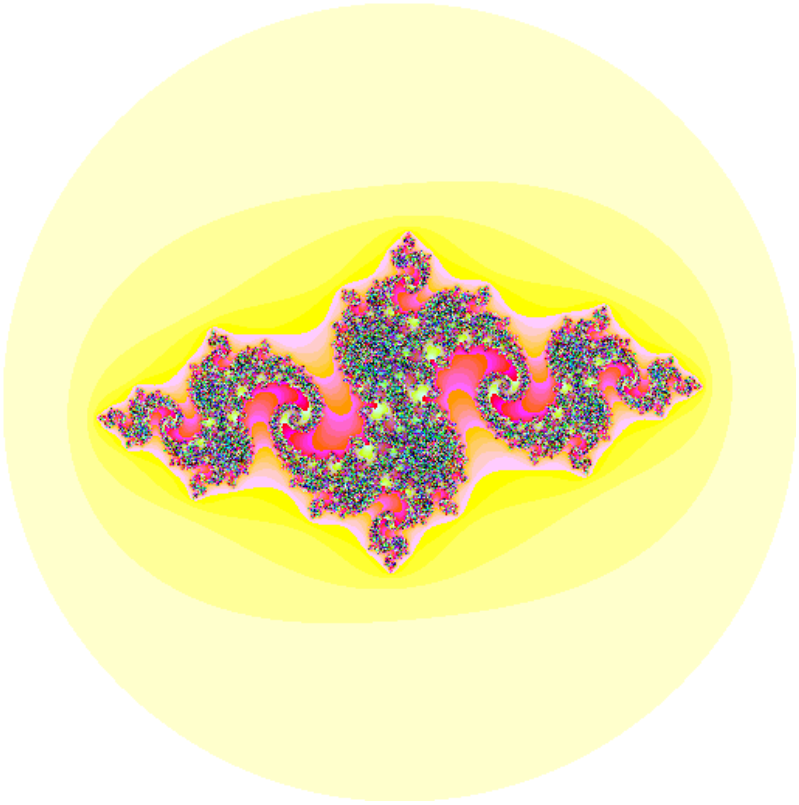
Donde z es una variable compleja y c una constante compleja. Se denomina conjunto de Julia de la anterior función al conjunto de valores complejos z que estabilizan su valor (no tienden a infinito).

Una forma de representar el Conjunto de Julia de la anterior función para un valor de c, consiste en aprovechar que el espacio complejo es bidimensional para representar el resultado de su evaluación en forma imagen. A los puntos cuyo valor se estabiliza (pertenecen al conjunto) se les asigna un color (negro) y a los que divergen otro (blanco). Para decidir que el valor es estable se establece un número máximo de iteraciones. Además, puede aprovecharse que los valores complejos del módulo $|z|>2$ no pertenecen al conjunto, por lo que si se alcanza este valor se puede detener la evaluación. Así queda una imagen como la siguiente:



Una forma más vistosa de dibujar los Conjuntos de Julia (y en general los fractales) consiste en emplear colores para representar la velocidad a la que los puntos que no pertenecen al conjunto escapan hacia el infinito.

Con esta segunda técnica se pueden obtener resultados tan vistosos como los que se presentan debajo. Jugando con la paleta de colores, el zoom y el encuadre, se pueden obtener resultados como el siguiente para un valor de c=-0,8+0.156i.



**Algoritmo secuencial**:

En esta práctica vamos a trabajar con el algoritmo que genera el Conjunto de Julia. Para ello, en un fichero que os podréis descargar en el swad (conjuntoJulia.c), se proporciona la versión secuencial del algoritmo. En él se definen algunas constantes como el tamaño de la imagen a generar (Fil, Col), donde comienza la ventana del espacio (Ventx, Venty), sus dimensiones (Tamx, Tamy) y el valor de c. Todos estos valores se encuentran como constantes en el código secuencial, para la versión MPI, se deberán pasar como parámetros de entrada.

Al ejecutar el algoritmo se genera un fichero llamado Salida compuesto de Fil x Col bytes que representan lo colores del Conjunto de Julia. Para generar un fichero de salida.gif que se pueda visualizar fácilmente, utilizar el siguiente comando:

**raw2gif -s 512 512 –p vgaPalette.txt salida.raw > salida.gif**

Donde Fil y Col se sustituirán por los valores correspondientes que se encuentran en el fichero del algoritmo. vgaPalette.txt es una paleta de 256 generada empleando la utilidad gifclrmp y la base de la paleta estándar VGA. Este fichero también está disponible en swad.

## *Duración*

La duración de esta práctica serán todas las sesiones restantes del cuatrimestre.

## *Entrega*

La entrega se realizará mediante defensa en el aula de prácticas. Para la entrega el alumno debe demostrar al profesor que sabe compilar y/o ejecutar el programa con MPI, que el programa funciona y que habéis tomado las medidas adecuadas. El alumno debe explicar cómo ha tomado las medidas de tiempos y cómo ha calculado las ganancias que se incluyen en las recomendaciones de trabajo. También debéis saber explicar cómo habéis resuelto el problema que se os plantea en esta práctica, qué estilo de programación habéis seleccionado y cómo están organizadas las comunicaciones y en general la gestión paralela del algoritmo. No se deberá entregar nada en papel y no se debe enviar nada por correo electrónico.

## *Recomendaciones de Trabajo*

1. Se recomienda analizar el problema mediante el algoritmo secuencial que acepte parámetros de configuración inicial en la línea de comandos del *shell* como las dimensiones de la matriz, y la partición de los intervalos. Todo lo que deba ser fijo para el funcionamiento del algoritmo debería quedarse como constante, pero lo que pueda variar, debe ser introducido como parámetro de entrada.
2. Realizar un análisis del problema intentando averiguar las tareas que pueden ser paralelizables, analizando las dependencias y estableciendo el tipo de paralelización más adecuada para una máquina de memoria distribuida como la que podéis montar en el laboratorio.
3. Trabajar sobre el algoritmo secuencial como punto de partida para vuestro algoritmo paralelo.
4. Implementar una solución paralela inicial con MPI tomando medidas de tiempos de ejecución utilizando diferente número de máquinas que debéis ir variando mediante los parámetros de ejecución.
5. Los tiempos de cálculo de diferentes Conjuntos de Julia pueden ser muy diferentes. Para evitar que las medidas diverjan demasiado, se fijará un valor de c (el que más le guste) y un encuadre (espacio complejo a evaluar) y se mantendrá durante todo el experimento.
6. Comprobar que los datos obtenidos son los correctos mediante la representación gráfica del Conjunto de Julia. Dicha visualización se puede hacer por programa (para aquellos que conozcan como se trabaja con gráficos) o usando alguna herramienta gráfica como el comando raw2gif o cualquier otro que queráis. La solución gráfica obtenida por el código secuencial debe ser similar a aquella obtenida por el código paralelo.
7. Identificar los problemas de vuestro algoritmo y realizar los cambios adecuados para optimizar el código de la versión paralela anterior tomando de nuevo medidas para averiguar la ganancia que ha supuesto dicha optimización.
8. Para proporcionar una medida de las prestaciones fiable, se realizarán medias de las mediciones realizadas (de al menos tres ejecuciones) y se presentarán las ganancias obtenidas debidas a la optimización de código y a la escalabilidad del sistema.
9. La práctica se defenderá con el profesor de prácticas. El alumno debe aportar para la defensa tanto material como estime necesario para demostrar el trabajo que ha realizado (esto incluye, gráficas de ganancia, tablas de resultados, información de cómo y en qué computadores ha realizado las medidas, etc).