Instituto Politécnico Nacional

Escuela Superior de Cómputo

Aplicaciones para Comunicaciones en Red

Profesor: Josué Rangel González

Práctica 1: Saturación de imágenes y Multiplicación de matrices implementando el uso de hilos

Alumnos: Camacho Pérez Karen Fernanda

Osornio Zambrano Alberto Aacini

Rodríguez Rios Jesús Iván

3CV7

Índice

[Introducción 3](#_Toc55228677)

[Desarrollo 4](#_Toc55228678)

[Diagrama de flujo 6](#_Toc55228679)

[**Flujo saturación de imágenes.** 6](#_Toc55228680)

[**Flujo multiplicación de matrices.** 7](#_Toc55228681)

[Conclusiones 7](#_Toc55228682)

[Camacho Pérez Karen Fernanda 7](#_Toc55228683)

[Osornio Zambrano Alberto Aacini 7](#_Toc55228684)

[Rodríguez Rios Jesús Iván 7](#_Toc55228685)

[Resultados 8](#_Toc55228686)

# Introducción

**Hilos**

Son de las menores estructuras lógicas de programación que se ejecuta de forma secuencial por parte del planificador del sistema operativo. Tienen un ciclo de vida que es desde su creación hasta que termina su objetivo, en el cual va pasando por diferentes estados

Nacido: cuando se acaba de crear un hilo, no se han asignado recursos del sistema todavía para el hilo.

Listo: cuando el método de inicio se invoca del hilo se dice que está en estado listo.

Ejecutable: crea los recursos del sistema para ser ejecutado el hilo, pero aún no ha empezado a ejecutarse el hilo.

En ejecución: los hilos de alta prioridad se ejecutan primero esto es cuando ya se le asigno los recursos.

No ejecutable: deja de ejecutarse cuando alguno de estos eventos sucede sleep, suspend, wait, bloqueado por una espera de una E/S.

Muerto: un hilo pasa al estado de muerto cuando termina ya sea por natural o provocado.

Saturación de imágenes.

La saturación es la pureza de un color. La saturación es un aspecto muy importante en la fotografía, quizás tan importante como contraste. Además de que nuestros ojos se sienten naturalmente atraídos por los tonos vibrantes, los colores tienen su propia forma única de contar una historia que juega un papel crucial en la toma de una fotografía.

Ya sabemos que cada color que vemos se traduce en un valor integral de 0 a 255, correspondiente a los tres colores primarios aditivos: Rojo, verde y azul (también conocido como RGB). Para que un color sea rojo puro, su valor en forma digital es (255, 0, 0). Para el verde puro, es decir (0, 255, 0), mientras que el azul puro es (0, 0, 255). Dado que estos colores no tienen ningún otro mezclado y tienen el valor RGB máximo de su color correspondiente se dice que están en la saturación máxima.

**Matrices**

Una matriz es un arreglo bidimensional de números y se utilizan para múltiples aplicaciones y sirven, en particular, para representar los coeficientes de los sistemas de ecuaciones lineales o para representar transformaciones lineales dada una base. En este último caso, las matrices desempeñan el mismo papel que los datos de un vector para las aplicaciones lineales.

La multiplicación de matrices consiste en combinar linealmente dos o más matrices mediante la adición de sus elementos dependiendo de su situación dentro de la matriz origen respetando el orden de los factores.

En otras palabras, la multiplicación de dos matrices es unificar las matrices en una sola matriz mediante la multiplicación y suma de los elementos de las filas y columnas de las matrices origen teniendo en cuenta el orden de los factores.

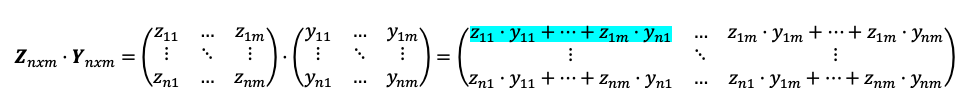


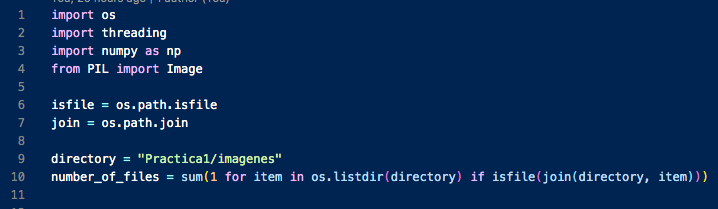
Ilustración 1. Procedimiento multiplicación de matrices

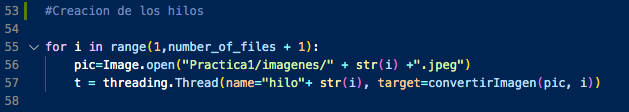
# Desarrollo

1. **Saturación de imágenes.**

Dentro de un directorio copiar más de 10 imágenes, cada una de estas imágenes deberán ser asignadas a un hilo. El hilo deberá a abrir la imagen como una matriz RGB y tendrá que generar 3 copias de dicha imagen, la primera copia asignará el valor R a 255, en la segunda copia asignará el valor G a 255 y en la tercera copia asignará el valor B a 255. Cada una de las copias modificadas deberán ser guardadas nuevamente como una imagen, lo cual provocará que al final se tengan por cada imagen original 3 más (una saturada en Red, otra saturada en Green y otra saturada en Blue).

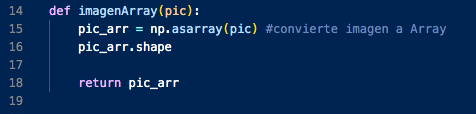
Para este programa el lenguaje utilizado fue Python, debido a la biblioteca PIL que facilita la manipulación de imágenes.



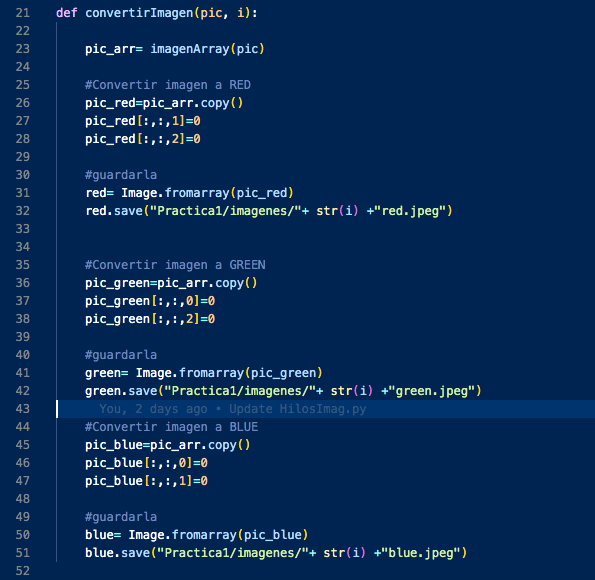


Para la transformación de la imagen a sus canales RGB se utilizaron dos funciones.

La función imagenArray() recibe la imagen y la convierte en un array bidimensional, para después regresarlo.



En convertirImagen() se manda a llamar a imagenArray() para poder tener la imagen como una matríz y después se hace la descomposición de esta en los tres colores, seleccionando sólo los valores que corresponden a cada color. Se vuelve a convertir la matriz a una imagen, guardándose con el nombre original concatenado con el color que le corresponde.



1. **Multiplicación de matrices.**

Generar la multiplicación de matrices donde A = 2000 \*1500 y B= 1500 \* 2000. Se deberán ejecutar diferentes cantidades de hilos que resuelvan esta matriz, en cada ejecución se deben medir los tiempos y al final se deberá generar una conclusión: ¿Qué sucede cuando más hilos ejecutan la matriz, vale la pena crear grandes cantidades de hilos para resolver el problema? La cantidad de hilos a ejecutar son: 2,4,8,16,32.

Se inicia declarando las 3 matrices matrizA y matrizB serán las que se multipliquen en ese orden y matrizC  donde se guarde el resultado por lo tanto esta matriz tendrá las dimensiones de las filas de matrizA y las columnas de matrixB.

typedef *struct* Matrix

{

*int* n;//filas

*int* m;//columnas

*int* \*\* contenido; //apuntador a la matirz

*char* nombre; //nombre de la matriz

}Matrix;

Matrix matrixA;

Matrix matrixB;

Matrix matrixC;

Posteriormente se le pide las dimenciones de las matrices A y B al usuario las setea con valores aleatorios y muestra el resultado

leerMatriz(&matrixA,'A'); //creae las matrices

leerMatriz(&matrixB,'B');

mostrarMatrix(&matrixA); // mostrar las matrices

mostrarMatrix(&matrixB);

si la multiplicacion es posible crea la matrizC con las dimenciones que se indico en A y B y pregunta por cuantos procesos se introduciran

if(matrixA.m == matrixB.n){ //verificando si es posible hacer la multiplicacio

//creando la matriz C

matrixC.nombre='C';

matrixC.n= matrixA.n;

matrixC.m=matrixB.m;

matrixC.contenido = crearMemoriaVacia(matrixC);

// preguntar el numero de hilos

printf("Ingrese el numero de hilos:");

scanf("%d",&noHilos);

Calcula cuantas filas procesara cada hilo y apartir de cual iniciara. Todos esto se lo pasara por parametros de uns estructura

//estructura para los parametros a los hilos

typedef *struct* dato{ //parametros para el hilo

*int* initFila;

*int* noFilas;

*int* analizado;

} ParametroComplejo ;

Se crea la cantidad de hilos especificados por el usuario, aunqeu si hay mas hilos que filas a procesar algunos hilos no realizaran ninguna tarea y se crearan inecesariamente. Posteriormente se espera a que acaben su ejecucion todos

for(*int* i=0; i<noHilos;i++){

if( 0 != pthread\_create(&threadsArr[i], NULL, thread\_routine, &(paramArray[i]) ) ){ //lanzar los hilos

printf("no se pudo crear hilo \n");

return -1;

}

}

for (*int* i = 0; i < noHilos; i++) // esperar por los hilos lanzados

{

ParametroComplejo \*salida; // resibir los resultados

pthread\_join(threadsArr[i], (*void* \*\*) &salida );

}

En el hilo se empezara a analizar desde la fila indicada y el numero indicado de filas.

for(*int* h=0;h<noFilas; h++){ // para cada una de las filas a analizar

for (*int* i = 0; i < matrixA.m ; i++){

for (*int* j = 0; j < matrixB.m ; j++){

matrixC.contenido[ filaActual ][j ] +=matrixA.contenido[ filaActual ][i] \*matrixB.contenido[ i ][j];

}

}

filaActual++; // pasar a la sigueinte fila a analizar

}

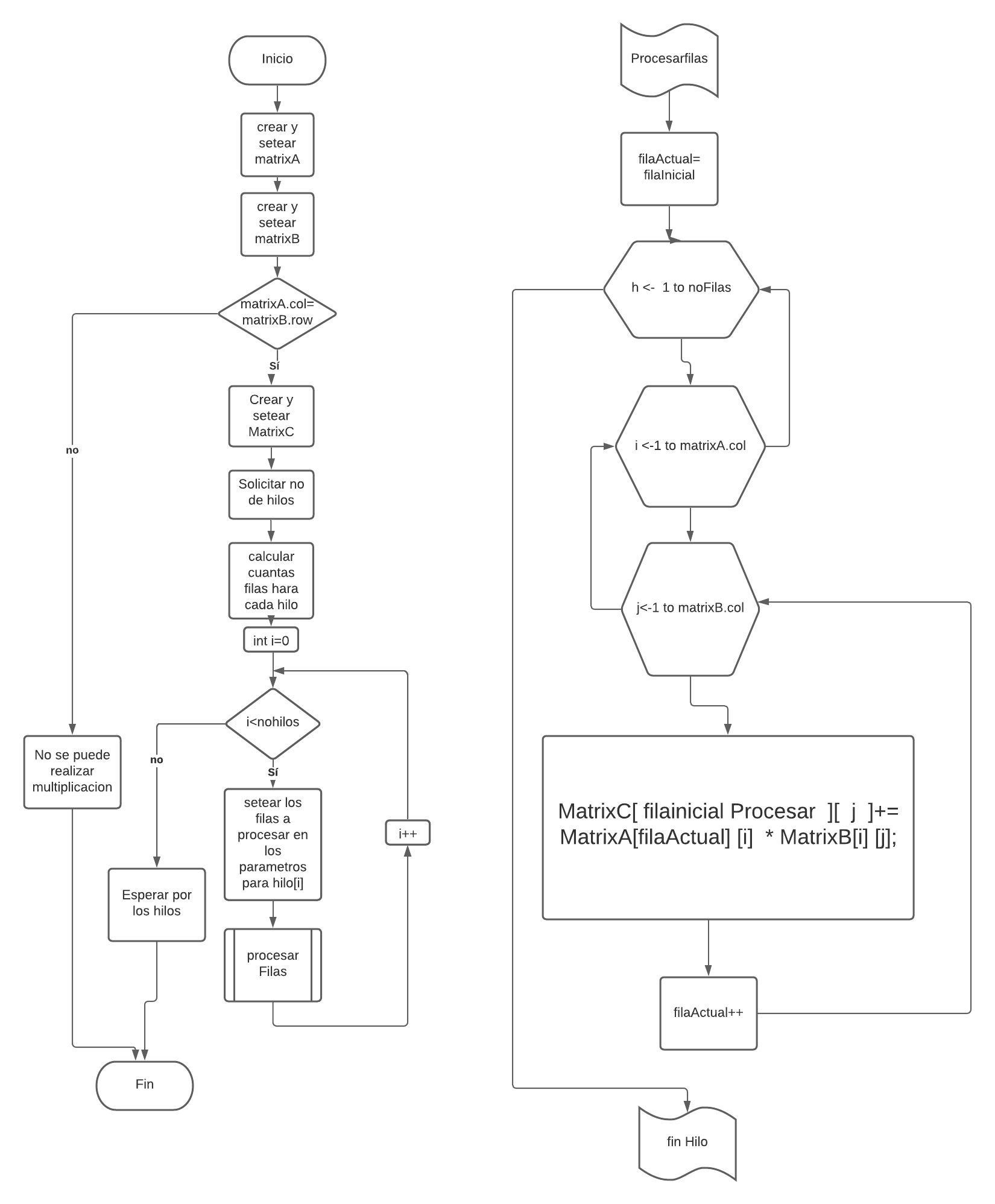
# Diagrama de flujo

## **Flujo saturación de imágenes.**

Diagrama

Descripción generada automáticamente

## **Flujo multiplicación de matrices.**



# Conclusiones

## Camacho Pérez Karen Fernanda

En esta práctica pudimos entender mucho mejor el funcionamiento de los hilos y cómo es que son una herramienta útil dentro del desarrollo y ejecución de una tarea, pues por medio de la concurrencia podemos lograr hacer una o varias tareas de forma mucho más rápida que si la hiciéramos solamente utilizando un proceso. Algo importante es que vimos la implementación de estos hilos en dos lenguajes de programación diferentes, C y Python, cuyo manejo de memoria y nivel son muy diferentes entre sí, pero aun así se puede concluir que en ambos se logró alcanzar los resultados deseados.

## Osornio Zambrano Alberto Aacini

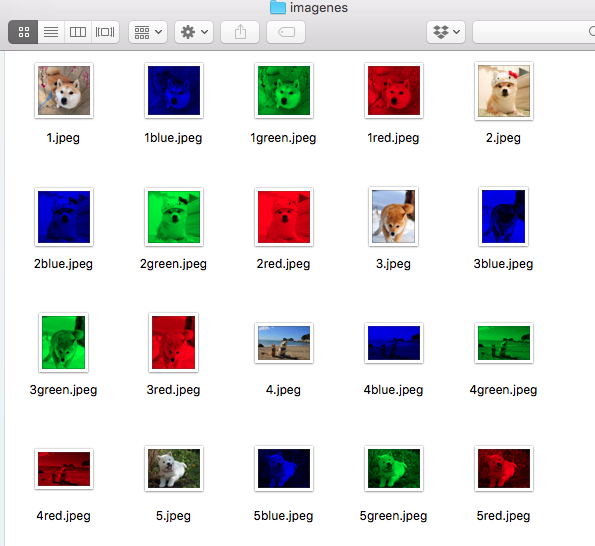
En esta practica pude ver la utilidad de los hilos para la programación , pero también vi los limites que tiene, no necesariamente por hacerlo con mas hilos el tiempo de ejecución se reducía inclusive hubo un caso donde teniendo el doble de hilos que la iteración anterior el tiempo de ejecución aumento, Esto se podría entender dado que aunque el problema se divide el numero de nucleos de la computadora es el mismo y a la computadora le toma mas tiempo crear los hilos y sigue teniendo la misma velocidad de procesamiento, ósea solo le estamos aumentando la carga de trabajo al ponerla a hacer mas hilos, incluso corriendo un hilo por fila a procesar el tiempo de ejecución no fue mejor que haciéndola con 2 hilos, que es el numero de nucleaos que tiene mi equipo, concluyo que es mejor lanzar el numero de hilos que es capaz de soportar nativamente el procesador

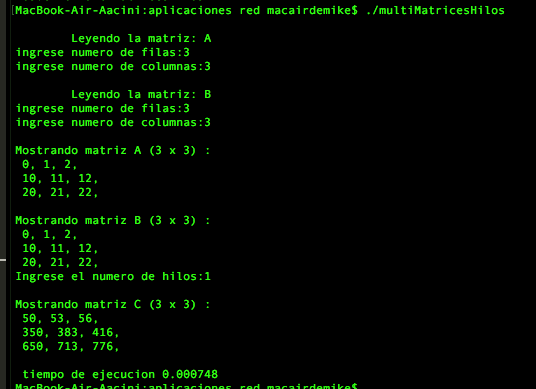
## Rodríguez Ríos Jesús Iván

El desarrollo de ambos ejercicios permitió que entendiera de mejor forma el funcionamiento e implementación de los hilos, el haber ocupado dos lenguajes diferentes fue beneficioso debido a que aprendí a implementar los hilos en cada lenguaje y así comprender un poco más a lo sistemas operativos que hacen uso de hilos para permitir que una aplicación pueda realizar varias tareas a la vez. La principal ventaja si se realiza bien la programación de los hilos, es que los programas operan con mayor velocidad, ya que los hilos del programa están orientados para la ejecución concurrente.

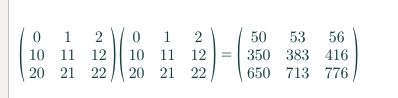
# Resultados

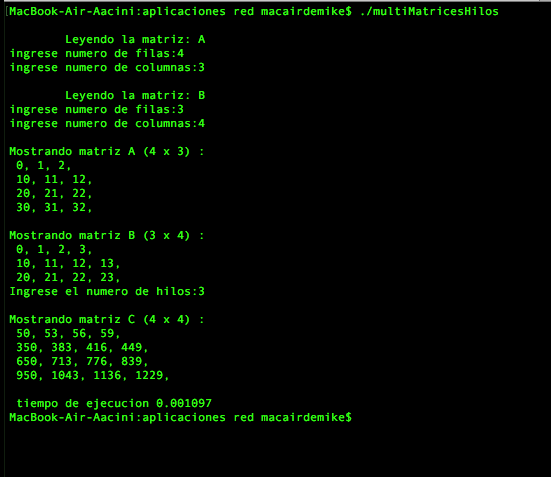
**Saturación de imágenes**



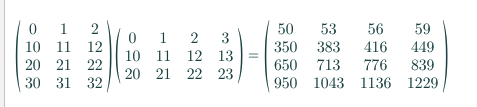
Multiplicación de matrices (comprobación de funcionamiento)

Matriz C 3x3 , 1 hilo





Multiplicación (4x3) \*(3x4), 3 hilos



Multiplicación (2000x1500) \*(1500x2000) (pueba de tiempo )

|  |  |
| --- | --- |
| 2 hilos, tiempo 64,913 segundos | 4 hilos tiempo 94.00 segundos |
| 8 hilos , 99 segundos | 16 hilos, 96.40 segundos |
| 32 hilos , 91.486 segundos | 2000 hilos, 93.44 segundos |

Conclusión

A pesar de que en unos casos el numeros de hilos se duplico el tiempo de ejecucion aumento, no se si en ese momento otra aplicación ejecuto un subproceso haciendo que la ejecucion aumentara de tiempo pero aun asi en el los tiempos donde decendia no fue por mucho tiempo aun teniendo el doble de hilos, inclusive por curiosidad ejecute con 2000 hilos , un hilo por cada fila de la matriz resultante y el tiempo apenas se vio mejorado, Esto parece indicar que aunqeu el numero de hilos aumenta no lo hace el numero de procesadores que tiene el equipo al cual le toma tiempo crear el hilo y ejecutarlo