

Universidad Nacional Autónoma de México

Facultad de Ingeniería

Computo de alto desempeño

Profesor: M.I. Rene Oscar Valdez Casillas

Equipo: Ponce de León Súchil Daniel

Gómez de la Cruz Luis Octavio

Semestre: 2020-1

Índice de Contenido

[Objetivos 2](#_Toc25604736)

[Problema 3](#_Toc25604737)

[Análisis del problema 3](#_Toc25604738)

[Introducción 3](#_Toc25604739)

[Desarrollo 3](#_Toc25604740)

[Análisis y Diseño: 3](#_Toc25604741)

[Funciones de MPI 3](#_Toc25604742)

[Análisis de Speed-up 3](#_Toc25604743)

[Manual de Uso del programa 4](#_Toc25604746)

[Conclusiones Generales 4](#_Toc25604747)

[Conclusiones Personales 5](#_Toc25604748)

Índice de Tablas

[Tablas 4](#_Toc25604762)

Índice de Graficas

[Tablas 4](#_Toc25604803)

[Graficas 4](#_Toc25604804)

## Objetivos

Analizaremos, desarrollaremos e implementaremos un programa de forma paralela.

Identificaremos en qué casos o circunstancias óptimas para implementar el computo en paralelo con las herramientas adecuadas en este caso se usar MPI

## Problema

Se requiere realizar un programa en ANSI C en su versión serial y paralela que lea una imagen PNM a color y la transforme a una imagen en escala de grises asi como también a negativo.

## Análisis del problema

Es esencial la versión del programa de forma serial para poder dar inicio a la transformación o la implementación de la paralelización, se debe conocer primeramente la estructura de la imagen PNM asi mismo de como transformar la información de la imagen para convertirla a escalas de Blanco Negro y Negativo, con la ayuda del “número mágico” se podrá realizar esta conversión.

## Introducción

MPI (Message Passing Interface) es una interfaz de paso de mensajes que representa un esfuerzo prometedor de mejorar la disponibilidad de un software altamente eficiente y portable (S.Pacheco, 1997). En la computación de alto rendimiento a través de la definición de un estándar de paso de mensajes universal, incluye interfaces principalmente para FORTRAN, C, C++.

MPI agrupa los procesos implicados en una ejecución paralela en comunicadores, que agrupa a procesos que pueden intercambiarse mensajes. (García, 2018)

## Desarrollo

### Análisis y Diseño:

Debemos de analizar el código serial para localizar que sección se puede implementar el paralelizado, en este caso se debe paralelizar en donde se vaya a tener más tiempo de ejecución en este código la mayor carga son las cargas y escritura de las matrices este es un punto clave para poder optimizar el proceso. Se opto en hacer una descomposición de dominio para repartir equitativamente o el más cercano el peso de procesamiento del código en este caso es más preferible la otra forma de distribución, pero teniendo en cuenta el tamaño de la imagen que puede ser mucho mayor y la otra forma podría ser no la más optima

### Funciones de MPI

(Moreira, s.f.)

### Análisis de Speed-up

#### Numero de procesos:2

##### Tamaño de imagen

##### Tamaño de imagen

#### Numero de procesos:4

##### Tamaño de imagen

##### Tamaño de imagen

#### Numero de procesos:6

##### Tamaño de imagen

##### Tamaño de imagen

#### Numero de procesos:8

##### Tamaño de imagen

##### Tamaño de imagen

## Tablas

**(Aquí van tablas de proceso vs tiempo)**

## Graficas

**(Aquí van las graficas de las tablas)**

### Manual de Uso del programa

1. Iniciamos el sistema operativo donde se tenga implementado la herramienta MPI, en este caso se usó Linux, distribución Ubuntu.
2. Ejecutamos la consola(terminal), nos posicionamos en la dirección donde se encuentre el código fuente con la respectiva imagen de base
3. Deberíamos tener 3 archivos
   1. El serial(pnmserial.c)
   2. El paralelo(pnmparalelo.c)
   3. La imagen(lena\_color)
4. Compilaremos los dos programas .c con respectivas instrucciones
   1. gcc pnmserial.c
   2. mpicc pnmparalelo.c -o pnmparalelo
5. Después de que el copilado se haya realizado es momento de ejecutar con los siguientes comandos respectivamente (# número de procesos debe ser entero mayor que 0)
   1. ./pnmserial
   2. Mpirun -np # ./pnmparalelo
6. Posteriormente verificaremos y abriremos las nuevas imágenes generadas
   1. Lena\_color-gray.pnm
   2. Lena\_color-neg.pnm

## Conclusiones Generales

Como podemos comprobar en las gráficas aumentando el número de procesos es más rápido la ejecución de las sentencias que se paralelizaron, pero no en todos los problemas computacionales no es recomendable hacer uso de la paralelización, y en otros casos no abrumar con el número de procesos esto puede entorpecer la operación.

## Conclusiones Personales

##### Luis Octavio Gómez de la Cruz

Fue un poco complicado controlar o hacer que la comunicación de los hilos o procesos ya que en cierto punto solo se realizaba una cuarta parte de la transformación de la imagen correctamente y el resto no llego o no se sobrescribió, este fue el mayor problema que presentamos

##### Daniel Ponce de León Súchil

El programa serie fue mas fácil de implementar, ya que consistía principalmente en una lectura ya esperada de un archivo y el manejo de matrices. El único problema fue el manejo de archivos grandes y las implicaciones que tenia en memoria. Para el programa paralelo se complicó la parte de administración de procesos y el manejo de las comunicaciones entre ellos.

Bibliografía

García, E. K. (2018). Introduccion a MPI. *Computo de Alto Desempeño*, 33.

Moreira, B. C. (s.f.). *MPI desde C*. Obtenido de http://www.eead.csic.es/compbio/material/programacion\_rocks/node9.html

S.Pacheco, P. (1997). *Parallel Programming with MPI.* EUA: Editorial and Sales Office.