UNIVERSIDAD
DE SANTIAGO
DE CHILE

Departamento de Matemática y Ciencia de la Computación

Alogritmos Distribuidos
Trabajo número 3

Segundo Seméstre 2022

04 de Diciembre del 2022

Florencia Céspedes Armella
Alexis Bolados del Castillo

Algoritmos Distribuidos
Licenciatura en Ciencia de la Computación

Ruben Carvajal Schiaffino
florencia.cespedes@usach.cl
alexis.bolados@usach.cl

Contenidos

1. Introducción

1.1.

Especificaciones
La especificaciones de este programa es crar un algoritmo distribuido de memoria privada que permite la suma o multiplicación de dos matrices.
La ejecución del programa se inicia con el siguiente comando:
mpirun -np K ./t3.exe -p -o -m ¡data.txt

- Con -np indicando el número de proceso.
- K el número de procesos.
- -p el modo de partición H, V Con H si es horizontal y V si es vertical.
- -o la operación S para suma y M para multiplicación.
- -m el modo de ejecución S, V Con S para silencioso y V para verbose.

El orden de las matrices es extraido de las cuatro primeras lineas del archivo data.txt, siendo los demás datos los que van dentro de ambas matrices.

1.2. Librerias

Las librerias implementadas para el programa son las siguentes :

- $< stdio.h >$ Para recivir y entregar información.
- $< stdlib.h >$ Archivo de cabecera de bibliteca en C.
- "$mpi.h$" Para el paso de mensajes.

2. Procedimiento

Presentamos la logica de desarrollo para la solución del problema.
Iniciamos reciviendo los parametros desde la terminal, usando la función **scanf()** para sacar los primeros cuatro digitos del archivo data.txt, los cuales son el orden de ambas matrices. Y la función **strcmp()** para saber que modo de partición, operación y modo de ejecución fue ingresado.
Tambien inicializamos los procesos con $MPI_Init()$ junto con el tamaño y en ranking, mostrando en pantalla los procesos vivos.
Se llama la la función MASTER, la cual maneja la distribución de todos los demás procesos SLAVE. Dependiendo de los parametros ingresados el programa se va a distintas funciones. La función SUMA se divide en 2 funciones: sumaVert, para las sumas de partición vertical y sumaHoriz para la suma de partición Horizontal.
En las funciones se mandan las dimensiones o los arreglos a trabajar por cada proceso slave.

2.1. Structs y Variables Globales

El algoritmo en sus inicios cuenta con 1 struct: MATRIX.

atributos	definicion
nRows	número de filas de la matriz
nCols	número de columnas de la matriz
data	doble puntero que guarda los datos

Las variables globales son varias, por ejemplo: 6 variagles globales son del tipo TAG, las cuales corresponden a las cantidades de mensajes que se realizan entre distintos **MPI_Send()** y **MPI_Recv()**.
Se crea desde el inicio una distinción entre los parametros de entrada, supongamos que se ingresa una V, indicando que la partición de la matriz debe ser de forma vertical, como variable global VERTICAL tiene valor 1 y HORIZONTAL 0; esto se mantiene de manera similar con los demás parametros de entrada.
Donde se guardan dichos datos señalados arriba tambien son variables globales: partition para la forma de partición, operation para la operación que va a realizar la matriz y mode para saber si el programa corre de modo silencioso o verdose.
Ambas matrices y una tercera son designadas como globales para que todas las funciones puedan acceder a ellas, como tambien sus ordenes.

3. Álgoritmo

Para acceder al algoritmo desde la consola es necesario escribir:
mpirun -np K ./t3.exe -p -o -m data.txt.
Siguiendo las especificaciones del trabajo K es la cantidad de procesos, p el modo de partición ($H.V$), -o la operación (S, M), -m el modo de ejecución ($S.V$) y data.txt el archivo que guarda los datos de las matrices.
En el **void main** entran como argumentos los datos indicados y escos se extraen con la función **scanf()** y luego se crean los hilos y se crean los usuarios. Despúes el programa trabaja con todos los datos que ingresan a las matrices.
Desde la función del master se crean todas las acciones, y como es un programa de memoria privada entre los hilos solo llegan copias de lainformación que necesitan trabajar. El master es quien indica que operaciones deben estar haciendo los hilos, ya sea suma vertical, horizontal, etc.
El master recive los datos que han trabajado los hilos y los junta en una misma matriz resultante aux, la cual se muestra al usuario al finalizar el programa.

4. Conclusión

Se ha presentado en este informe un programa que trabaja con mensajes entre procesos atraves de la libreria MPI y de memoria privada.
Una de las principales complicaciones vinieron a la hora de que los mensajes fueran bien recividos entre estos procesos, pues muchas veces los argumentos colocados en estos mensajes no eran los correctos.
En este trabajo no se incluye la multiplicación de las matrices pues se priorizo el buen funcionamiento de la funcion suma entre matrices.