****

**Pontificia Universidad Católica Madre y Maestra**

**Facultad de Ciencias de la Ingeniería**

**Asignatura:**

Programación II

**Asunto:**

Proyecto Final

“Programación Paralela Funciones MPI Send & Receive”

**Estudiantes:**

Eric T. Nuñez C. 2014-1329

Johanna R. Rodríguez R. 2014-1811

Félix Alejandro Guzmán 2014-0565

Joel Ant. Rodríguez 2012-1776

**Profesor:**

Ing. José L. Alonso

**Fecha de entrega:**

05 de Diciembre del 2016

**Modificar el programa solución del ejercicio 3.2 Send Receive del tutorial para que el proceso 0 difunda su identificador de proceso (0) al resto de procesos con identificadores pares, siguiendo un anillo de procesos pares, y el proceso 1 haga lo mismo con los procesos impares. Se deben tener en cuenta soluciones con cualquier número de procesos.**

Utilizando el ejercicio de programación en paralelo Send & Receive, cuyo funcionamiento consiste en que cada procesador le envíe su identificador al proceso siguiente, de manera que estos recibirán del proceso anterior su identificador, se nos adjudica la tarea de modificarlo, de manera tal que cada proceso par se encargue de enviar al siguiente par su identificador, al igual que los impares, que envían a los impares siguientes su identificador. Por ejemplo, si se trataran de 4 procesadores, el resultado en pantalla sería el siguiente:

**P2 recibe de P0**

**P3 recibe de P1**

De esta manera, y como se ha indicado anteriormente, nuestra tarea consistió en modificar el programa para que, tal y como indica el mandato, el identificador 0 se encargue de enviar al próximo proceso con identificador par, y este a su vez al siguiente par formando un anillo de proceso pares donde, por ejemplo:

**P0 envía a P2**

**P2 recibe de P0**

**P2 envía a P4 ….**

De igual forma, el código establece que el proceso con identificador 1 realice el mismo procedimiento, pero con el próximo proceso con identificador impar, y este a su vez le envía al siguiente completando un anillo de procesos impares donde, por ejemplo:

**P1 envía a P3**

**P3 recibe de P1**

**P3 envía a P5 ….**

En ambos casos, los procesos que inician la interacción son los procesos 0 y 1, que no reciben de ningún otro.

La lógica utilizada para modificar el código en lenguaje C++, haciendo uso de la librería MPI, y cumpliendo con todo lo requerido, consiste en una serie de sentencias de if y else que evalúan ciertas condiciones y, dependiendo de su resultado, se ejecutará una de las sentencias establecidas.

Básicamente, iniciamos con la condición que evalúa y establece si el proceso tiene como identificador 0. A continuación, este se encarga de enviar su Rank(identificador) a el siguiente con identificador par (0+2), y así mismo ocurre con el proceso con identificador 1 ,que envía al siguiente impar (1+2) su identificador. Por otra parte, ambos casos se aseguran de que se establezca un número válido de procesadores, suficientes para que el programa realice su función con éxito.

Luego se establecen las sentencias para que los siguientes procesos puedan recibir del proceso con identificador par o impar que lo antecede (su Rank-2) el identificador correspondiente. Comenzando con los identificadores pares que reciben, estos almacenan en una variable contadora lo recibido. A continuación, el procesador imprime en pantalla el identificador obtenido. Después de esto, se corrobora si el procesador par en cuestión puede enviar al siguiente par su identificador, es decir, si existe algún otro par después del actual que pueda recibir. Este encarga además de aumentar el contador en 2, para que el proceso actual siga enviando al próximo proceso par, siempre que haya procesos (size) faltantes. Este mismo procedimiento se repite para los procesos con identificadores impares.

El código esencialmente, establece qué proceso puede enviar y a cuál proceso envía, además, de cuál proceso puede recibir y de quien recibe, a la vez de establecer cómo seguir la cadena de envío y recepción entre procesos, sin que se produzca ningún caso en el que algún proceso envíe su identificador a otro no existente.

Para realizar esta práctica, principalmente se utilizó el recurso web: <http://lsi.ugr.es/jmantas/pdp/tutoriales/tutorial_mpi.php?tuto=02_send_receive> . Además, para el trabajo en equipo, se ha utilizado la herramienta Code Your Cloud, que permite modificar ficheros simultáneamente desde distintas computadoras.