Solucionario de la Práctica Calificada 01 de Programación Paralela

Datos del solucionario

Alumno	Garcia Moreno, Vieri Enrique	**Fecha elaboración**	05-02-20
Código	16200127	**Ciclo**	2020-0

1. **Explique con sus palabras ¿Qué es un proceso de una computadora?**

   Rpta. Es una serie o conjunto de instrucciones que hacen uso de un procesador.

2. **Explique a que se refieren cuando hablamos de una comunicación punto a punto entre 2 procesos, proponer un ejemplo en código.**

   Rpta. Una comunicación punto a punto es aquel tipo de comunicación entre procesos en la cual solo se involucran 2, uno encargado de enviar un mensaje y otro de la recepción del mismo. [Revisar programa: p2.sendrecv.cpp]

3. **¿Qué es una memoria RAM (principal), caché y virtual? E indicar cómo funcionan.**
   - **Memoria RAM**: es la memoria principal en la cual se pueden realizar operaciones de lecturas y escrituras.
   - **Memoria Caché**: es la memoria volátil, dedicada a los datos usados o solicitados con más frecuencia para su recuperación a gran velocidad.
   - **Memoria Virtual**: reserva un espacio en el disco duro, que simule la memoria RAM, en caso esta se encuentre saturada.

4. **¿En qué consiste la programación en Memoria Distribuida y la programación en Memoria Compartida?**

   Rpta. En la programación en memoria distribuida o multicomputador, cada unidad de procesamiento es totalmente independiente de las otras, contando cada una con un procesador y memoria propia, pero usándolas en conjunto para ejecutar un mismo programa; en el caso de la programación en memoria compartida, si bien las unidades de procesamiento trabajan también en ejecutar un mismo programa, estas no tienen memoria independiente por lo

que comparten una única memoria a través de algún bus de datos u otro mecanismo.

5. Describa en 3 líneas como máximo e indicar los parámetros de los siguientes comandos del MPI:

a) **MPI_Send**: rutina MPI usada en la comunicación punto a punto para enviar un mensaje de un proceso a otro proceso en específico que pertenezca al mismo comunicador.
Entre sus argumentos tenemos los siguientes: el buffer de datos a enviar, el tamaño del buffer, el tipo de dato de los datos del buffer, el identificador del proceso destino, una etiqueta de la comunicación y el comunicador común.

b) **MPI_Recv**: rutina MPI usada en la comunicación punto a punto para que un proceso pueda recibir un mensaje de otro proceso que al mismo comunicador.
Entre sus argumentos tenemos los siguientes: el buffer de datos a recibir, el tamaño del buffer, el tipo de dato de los datos del buffer, el identificador del proceso origen, una etiqueta de la comunicación y el comunicador común.

c) **MPI_Reduce**: rutina MPI usada para reducir información generada por procesos esclavos en uno resultado de aplicar alguna operación sobre todos estos el cual será obtenido por un único proceso maestro.
Entre sus argumentos tenemos los siguientes: el buffer de datos a enviar, buffer de datos a recibir, el tamaño de los buffers, el tipo de dato de los datos de los buffers, operación a aplicar, identificador del proceso destino y el comunicador común.

d) **MPI_Allreduce**: rutina MPI con funcionalidad similar a la anterior, con la diferencia de que en este caso todos los procesos del comunicador acceden al resultado de la reducción.
Entre sus argumentos tenemos los siguientes: el buffer de datos a enviar, buffer de datos a recibir, el tamaño de los buffers, el tipo de dato de los datos de los buffers, operación a aplicar y el comunicador común.

6. **Utilizando MPI, implemente un algoritmo que determine el número de veces que un elemento X aparezca en un vector A con n elementos entero. Se puede asumir que su algoritmo comienza con los elementos ya distribuidos entre los p procesos (n/p para cada uno).**

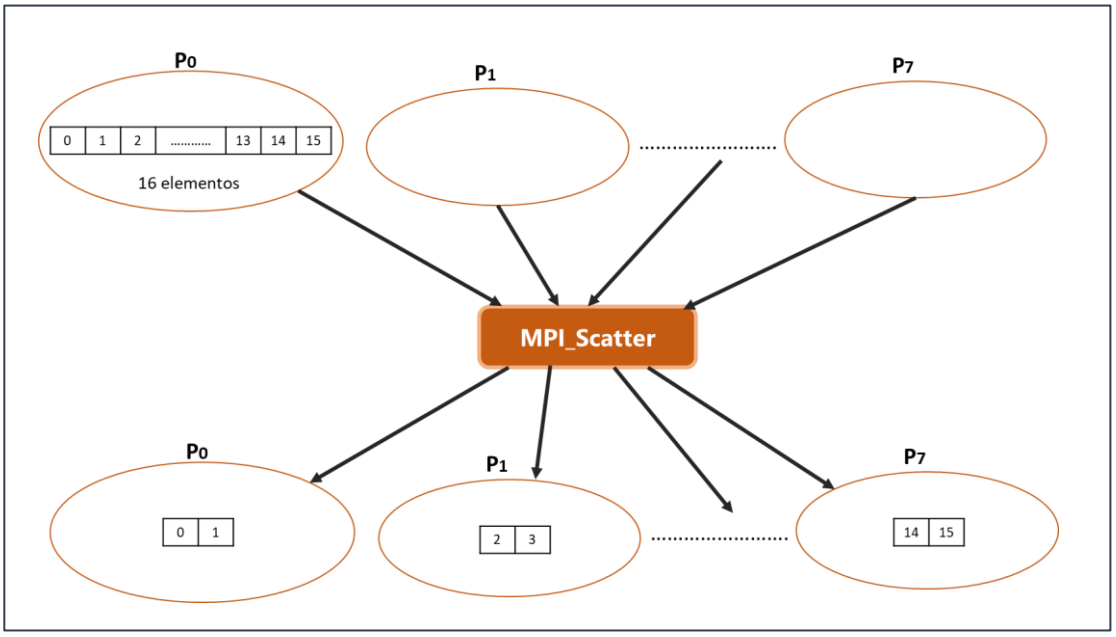
   Rpta. [Revisar programa: p6.contador.cpp]

7. **Desarrolle un algoritmo en MPI, utilizando p procesadores para calcular n! .**

   Rpta. [Revisar programa: p7.factorial.cpp]

8. **Suponga que comm_sz=8 y la cantidad de elementos es n=16.**
   a) **Diseñe un diagrama que explique como MPI_Scatter puede ser implementado usando comunicaciones basadas en árboles. Puede suponer que el origen del scatter es el proceso con rank 0.**

   Rpta.



b) Hacer lo mismo para el MPI_Gather, en este caso con el proceso 0 como destino.

Rpta.