**1° Practica Calificada del curso de Programación Paralela**

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

Un proceso es una ejecución de diversas instrucciones por parte del microprocesador, asociado a lo que indica un programa. El sistema operativo de la computadora es el que se encarga de gestionar estos procesos.

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

La comunicación punto a punto se realiza específicamente entre 2 procesos únicamente, donde un proceso envía un mensaje y el otro lo recibe. Además, este tipo de comunicación esta basada en etiquetas y comunicadores.

**Ejemplo**: Desarrollado en Eclipse Parallel Programming.

1. **¿Qué es una memoria RAM (principal), Cache y Virtual? E indicar ¿Cómo funcionan?**

La **memoria RAM** es la memoria de trabajo principal y temporal que está compuesta por direcciones de memoria donde se ejecuta la mayor parte del software. Cuando el procesador necesita realizar tareas de escritura o lectura, todas estas se llevan a cabo en la memoria RAM.

La **memoria cache** es un espacio de almacenamiento pequeño, pero rápido y exclusivamente dedicado a los datos usados con mayor frecuencia para su recuperación a gran velocidad. Cuando el procesador realiza tareas muy frecuentes de escritura y lectura, estas son almacenadas en la memoria cache para una recuperación mucho más rápida.

La **memoria virtual** es una técnica de gestión de memoria que se encarga de que el sistema disponga de mayor cantidad de memoria que esté disponible físicamente. Cuando queda poca memoria RAM, la memoria virtual mueve datos de la RAM a un espacio en el disco.

1. **¿En que consiste la programación en Memoria Distribuida y la programación en Memoria Compartida?**

La programación en **memoria distribuida** esta representada por bloques donde cada procesador cuenta con su propio espacio de memoria. Para realizar una comunicación es necesario el paso de mensajes a través de una interconexión

La programación en **memoria compartida** se representa como un conjunto de procesadores que tienen acceso a un único espacio de memoria a través de una red de interconexión. Donde cada procesador puede acceder a una misma dirección de memoria.

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

* **MPI\_Send** (void\* msg\_buf\_p, int msg\_size, MPI\_Datatype msg\_type, int dest, int tag, MPI\_Comm communicator): Método de MPI que envia un mensaje msg\_buf\_p, de un tamaño msg\_size y de tipo msg\_type, al proceso dest y con etiqueta tag a través del comunicador communicator.
* **MPI\_Recive** (void\* msg\_buf\_p, int buf\_size, MPI\_Datatype buf\_type, int source, int tag, MPI\_Comm communicator, MPI\_Status\* status\_p): Método de MPI que recibe un mensaje msg\_buf\_p, de un tamaño buf\_size y de tipo buf\_type, del proceso source y con etiqueta tag a través del comunicador communicator.
* **MPI\_Reduce** (void\* input\_data\_p, void\* output\_data\_p, int count, MPI\_Datatype datatype, MPI\_Op operator, int dest\_process, MPI\_Comm comm): Método de MPI que toma una entrada input\_data\_p y una salida output\_data\_p que operara con datos de tamaño count y tipo datatype con el método operator. La salida será retornada en el proceso dest\_process a través del comunicador comm.
* **MPI\_AllReduce** (void\* input\_data\_p, void\* output\_data\_p, int count, MPI\_Datatype datatype, MPI\_Op operator, MPI\_Comm comm): Método de MPI que toma una entrada input\_data\_p y una salida output\_data\_p que operara con datos de tamaño count y tipo datatype con el método operator. La salida será retornada a todos los procesos a través del comunicador comm.

1. **Utilizando MPI, implemente un algoritmo que determine el numero de veces que un elemento X aparezca en un vector A con N elementos enteros. Se puede asumir que su algoritmo comienza con los elementos ya distribuidos entre los P procesos (N/P para cada uno).**

Desarrollado en Eclipse Parallel Programming

1. **Desarrolle un algoritmo en MPI, utilizando P procesadores para calcular N!.**

Desarrollado en Eclipse Parallel Programming

1. **Suponga que COMM\_SZ = 8 y la cantidad de elementos es N = 16**

* Diseñe un programa que explique cómo **MPI\_Scatter** puede ser implementado usando comunicaciones basadas en árboles. Puede suponer que el origen del scatter es el proceso con rank 0.

El vector de 16 elementos es separado en 8 partes consecutivamente y en orden. Por ejemplo, en proceso con rank = 0 obtendra un vector con las posiciones 0 y 1 del vector original, el proceso con rank = 1 obtendra un vector con las posiciones 2 y 3 del vector original, y así sucesivamente.

* Hacer lo mismo para el **MPI\_Gather,** en este caso con el proceso 0 como destino.

Los vectores de cada proceso rank serán agrupados en único vector ordenado y enviado al proceso con rank = 0. Por ejemplo, El proceso con rank 0 recibirá un vector donde la posiciones 0 y 1 corresponde al vector que fue enviado por el rank 0, las posiciones 2 y 3 corresponden al vector que fue enviado por el rank 1, y así sucesivamente.