**Integrantes: Alvaro Zapata y Diego Delgado**

**Parte I (10 puntos)**

Responda las siguientes preguntas en 100 palabras o menos

1. (1 punto) ¿Qué es un proceso embarrassingly parallel y uno inherentemente serial? Dé un ejemplo de cada uno (distintos a los vistos en clase)

Un proceso embarrassingly parallel consiste en ejecutar tareas que pueden dividirse en partes independientes sin necesidad de comunicación entre ellas, como el procesamiento de imágenes en un gran conjunto de fotos. Por el contrario, un proceso inherentemente serial requiere una ejecución secuencial específica, como la ejecución de un algoritmo donde cada paso depende del resultado del paso anterior.

1. (1 punto) ¿Cuáles son los dos principales cuellos de botella al paralelizar un proceso? Explíquelo a partir de las leyes de Amdahl y Gustafson

Los principales cuellos de botella son la porción serial del programa, según la Ley de Amdahl, y la eficiencia de la paralelización que disminuye con el aumento de la porción paralelizable, según la Ley de Gustafson. La Ley de Amdahl dice que el tiempo de ejecución está limitado por la parte no paralelizable del programa, mientras que la Ley de Gustafson se enfoca en cómo la escalabilidad se ve afectada por la eficiencia de la paralelización.

1. (1 punto) Describa los recursos (CPU y GPU) de su computadora y provea evidencia (ie. screenshot).

No tengo GPU solo CPU.



1. (1 punto) ¿En qué se diferencia un CPU de un GPU? Dé un ejemplo de un proceso que convendría paralelizar en cada uno de estos tipos de unidad de procesamiento.

Un CPU está diseñado para realizar una amplia gama de tareas y es adecuado para operaciones seriales, como la ejecución de un navegador web. En cambio, un GPU está optimizado para tareas paralelas, como el renderizado de gráficos en videojuegos. Por ejemplo, sería optimo paralelizar el procesamiento gráfico (como videojuegos) en un GPU, mientras que la ejecución de algoritmos de búsqueda sería adecuada para un CPU.

1. (2 puntos) Se sabe estima que la nueva memoria RAM POWER 3000 permitirá acelerar las operaciones de su computadora en la mitad del tiempo. ¿Qué tanto más rápido será la velocidad de ejecución de un programa si el 77% del mismo se dedica a operaciones inherentemente paralelizables?

Si el 77% de un programa es paralelizable y la nueva memoria permite acelerar las operaciones a la mitad del tiempo, la aceleración total del programa se puede estimar usando la Ley de Amdahl.

Donde P es la fracción paralelizable y N es el factor de aceleración de la memoria.

1. (2 puntos) Piense en una tarea serial que le han encargado paralelizar. Describa el diseño de la implementación en paralelo de dicha tarea siguiente el método de Foster y los cuatro elementos que lo componen.

Si por ejemplo queremos ordenar una lista larga, se puede usar método de Foster. Primero, se puede dividir la lista en sublistas más pequeñas. Luego, se le asigna estas sublistas a diferentes procesadores. Despues, calcular la ordenación de las sublistas en paralelo. El último paso sería combinar las sublistas ordenadas en una lista final ordenada.

1. (2 puntos) ¿Cuál es la diferencia entre *point-to-point communication* y *collective communication* en MPI? ¿Cuál es la diferencia entre las operaciones de *broadcasting, gathering* y *scattering*?

La comunicación *point-to-point* implica la transferencia de datos entre dos procesos individuales específicos, utilizando funciones como `MPI\_Send` y `MPI\_Recv`. En cambio, la comunicación colectiva involucra a todos los procesos de un grupo y utiliza operaciones como `MPI\_Bcast`, `MPI\_Gather` y `MPI\_Scatter`, que afectan a todos los procesos simultáneamente. Broadcasting (`MPI\_Bcast`) permite que un proceso envíe el mismo dato a todos los demás procesos, gathering (`MPI\_Gather`) permite que cada proceso envíe sus diferentes datos al proceso raíz para consolidarlos, y scattering (`MPI\_Scatter`), lo contrario al Gather, distribuye segmentos desde el proceso raíz a los demás procesos para procesamiento paralelo.