Examen Segundo Parcial

Oscar Andrés Rosas Hernández, 417024956
Facultad de Ciencias. Introducción a la programación en paralelo con MPI, OpenMP y CUDA

I. OpenMP

- **¿Cuál es la filosofía detrás del modelo de funcionamiento de OpenMP?** La idea en un par de palabras es la de memoria compartida. Es decir, esta creado bajo la idea de que por ejemplo tienes varios nucleos o procesadores que pueden acceder a una región de memoria común.
  La idea de trabajo es bastante parecida a otros "frameworks", tienes un hilo maestro o controlador que orquesta a los demás hilos para que estos se repartan el trabajo y lo hagan de manera concurrente.
- **¿En propias palabras, ¿cuál es la principal ventaja de OpenMP, respecto a otras herramientas para programar en paralelo?.**
  De manera personal encuentro su modelo de pensamiento bastante mas claro que MPI por ejemplo; de manera mas clara la gran ventaja es que tenemos la habilidad de hacer nuestro algoritmo paralelo de manera incremental, es decir, podemos tener un codigo en secuencial e irlo pasando poco a poco a parelelo.

II. MPI

- **¿Cuál es la principal diferencia entre MPI y OpenMP?.** Volviendo al punto anterior, la idea es que MPI se basta en la idea de mensajes, es decir esta orientado a la idea de un cluster: muchos nodos que pueden realizar trabajo pero que no tienen un area comun de memoria.
  Ademas, para MPI tu tienes que pensar en tu algoritmo de manera paralela desde el principio, y requieres de muchas mas directivas (en general) que OPENMP.
- **Menciona ventajas y desventajas de MPI, justifica tu respuesta.**
  Ventajas:
  - Si tienes un cluster, es basicamente tu unica opcion :)
  - Si tu programa no depende de una gran cantidad de informacion (por ejemplo sumar integrales donde cada unidad de trabajo, la integral no requiere de mucha memoria para empezar a trabjar ni para mandar de regreso el resultado) entonces es una buena idea.
  - En general puede usarse en casos mas generales que OPENMP.
  - Cada proceso tiene sus propias variables de manera natural

  Desventajas:
  - Adios a la adopcion incremental de tu algoritmo
  - Por su misma naturaleza puede ser algo mas dificil encontrar bugs
  - Un nuevo cuello de botella ahora es la red
- **¿Cuál es la diferencia entre un hilo y un proceso?**
  Que buena pregunta, de manera general (sin ver mucho con MPI), un proceso tiene asociado un espacio de memoria propio, un proceso puede tener uno o mas hilos, un hilo o hilo de ejecucion de manera mas formal.
  Siendo mas puntual: Los procesos son independientes, tienen su propio espacio de memoria, por lo mismo son mas pesados para el sistema operativo, en general tanto crearlos como hacer cambios de contexto es mas tardado, ademas se comunican entre ellos por medios de los mecanismos que nos de el sistema operativo.
  Los hilos existen dentro de un proceso, tienen memoria compartida, en contraria es mucho mas rapido cambiar entre ellos (context switching), se comunican mediante variables compartidas.

III. CUDA

- **¿Que diferencía a CUDA de herramientas como OpenMP o MPI?** La idea creo que fundamental aqui creo que es que CUDA trabaja sobre cosas que no son las CPU, especialmente fue diseñada para trabajar en las GPU's (unidades de computo, no de proposito general sino especializadas en ciertas tareas), la idea es tomar tu informacion, enviarla al (las) GPU para ser procesada y regresarla, por lo tanto un cuello de botella aqui podria ser el envio de informacion.
  Tambien esta la idea de kernels y de pensar tu algoritmo como una serie de los mismos, algo tambien importante es que CUDA esta hecho un NVIDIA y no es un estandar abierto como OpenMP o MPI.

- **¿Por qué el uso de CUDA es ideal para desarrollo de algoritmos empleados en inteligencia artificial?** La verdad solo hablare de machine learning porque es la unica area de inteligencia artificial que se que usa en en exceso CUDA.
  Y la respuesta que me se, es que las redes neuronales y en general muchas areas de ML son estupidamente paralelizables, porque estamos realizando muchas operaciones que se pueden realizar de manera independiente, es decir podemos pensar en entrenar una red neuronal como muchas pequeñas operaciones que usualmente no dependen entre si.
  Ademas de trabajar con la idea de matrices (y tensores) o convoluciones, cosas para las que las GPUs son especialmente buenas paralelizando.
  Ademas para un programador es tan sencillo como `tensor([1., 5., 6], device='cuda:0')`

IV. VIDEOJUEGOS

La verdad hace muchos meses que no tengo tiempo de jugar :c, (incluso en la cuarentena) pero de los que de verdad me han gustado y que espero que a quien lea esto le sirva de recomendacion es:

- Si te gustan los libros de Asimov: Detroid become human
- Si quieres pasartela bien y con amigos y hacer muchos easter eggs muy raros pero con un lore muy interesante: Call of Duty: Zombies (de todos los Black Ops).
- He escuchado maravillas de Undertale, asi que creo que vale la pena echarle un ojo