Reporte GPU Para Procesamiento en Paralelo

Méndez Gutiérrez Néstor Javier

Ingeniería en Informática

Generación: 2015, Clave UASLP: 250980

Clave Fac. Ing. 201501400847

Materia: Supercomputo, Grupo: 222601

Fecha: 11 de noviembre de 2019

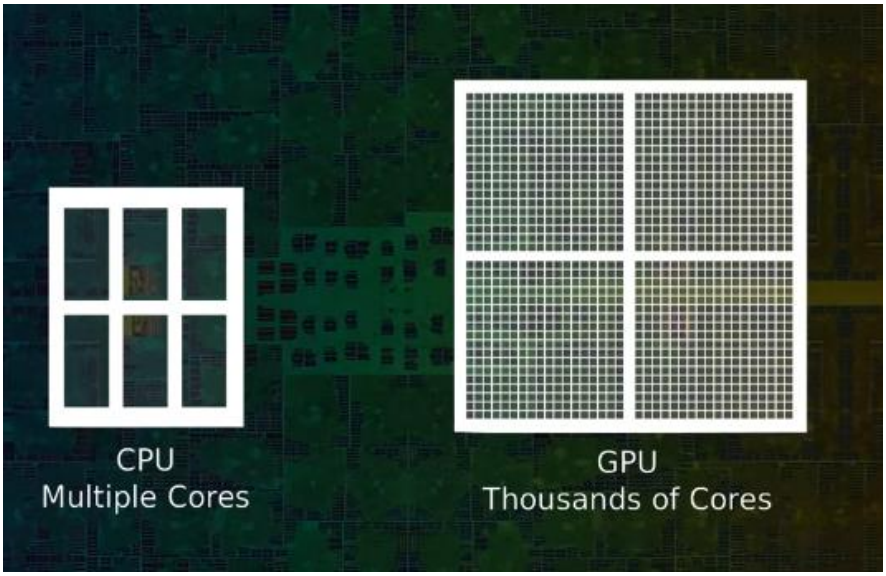
GPU + IA

Introducción

¿Por qué GPU y no CPU para Inteligencia artificial? En principio hay que aclarar que las CPU se les conoce también como procesadores de propósito general, esto nos habla de la versatilidad de tareas que puede ejecutar un CPU, desde procesar textos, enviar emails o hasta jugar Fortnite, pero esto no quiere decir que este tipo de procesadores este diseñado para ejecutar de la manera mas optima cada una de estas tareas, un ejemplo podría ser la implementación de una red neuronal que analice datos en tiempo real que generan algunos sensores, probablemente la implementación de una red neuronal de este tipo en un procesador de propósito general, sea bastante complejo, y es aquí donde el diseño de un chip con una arquitectura especializada que realice cálculos que requieran de una cantidad menor de ciclos de reloj nos puede ofrecer un rendimiento considerablemente mayor en comparación con un procesador de propósito general, el punto es que una arquitectura de hardware especializada puede traer beneficios que una CPU simplemente no puede ya que no esta diseñada especialmente para eso, en este caso las GPU y es que la GPU es un procesador de propósito especifico cuya especialización es la de procesar gráficos.

CPU vs. GPU

A grandes rasgos, la gran diferencia entre CPUs y GPUs seria la cantidad de núcleos de procesamiento que tiene cada una.



Y esto es muy importante ya que el numero de núcleos el cual es mayor por varias ordenes de magnitud en una GPU, nos permitirá la ejecución de un número mayor de instrucciones en paralelo, un clásico ejemplo del procesamiento en paralelo es la multiplicación de matrices, este tipo de procedimiento es muy usado en el procesamiento de imágenes, en donde dichas imágenes se componen de caras las cuales a su vez esta compuestas de vértices y estos a su vez se representan como vectores en un espacio tridimensional, es así como cada uno de los vértices que componen una imagen pueden ser procesados aplicando productos escalares que en este caso calcularían iluminación, texturas, sombreado, etc., al final un conjunto de tareas que por su naturaleza están muy preparadas para sacarle todo el provecho a el procesamiento multihilo que ofrece una GPU.

GPGPU

General Purpose on Graphics Processing Units: es el uso de tarjetas gráficas con propósito general y es que ha habido personas que se dieron cuenta que se podía echar mano de el paralelismo que ofrecen las GPU para poder resolver problemas que no necesariamente tienen que ver con el procesamiento de gráficos, ejemplo de ello son: computación de dinámica de fluidos, dinámica de partículas, modelado climático, finanzas computacionales, criptografía, minado de criptomonedas, renderizado de imágenes médicas, modelado de moléculas y muchas otras tareas y campos de estudio que se puedan ver beneficiados del paralelismo.

Machine Learning y GPUs

El machine Learning ha sido uno de los campos que se ha visto beneficiado por el uso de las GPUs, es por ello que cuando surgió la tendencia del GPGPU Nvidia en 2007 lanzo una plataforma que permitía usar las GPU y programarlas para resolver otro tipo de tareas, dicha plataforma es CUDA, el lanzamiento de CUDA comenzó a generalizar el uso del paralelismo para resolver otro tipo de tareas, posteriormente en 2009 se publicó **Large-scale Deep Unsupervised Learning using Graphics Processors**, articulo donde se presentan los resultados de entrenar una red neuronal haciendo uso del paralelismo que ofrece una tarjeta grafica en comparación con una CPU, los resultados arrojados por el articulo es que el uso de la GPU ofrecía una velocidad de entrenamiento para la red neuronal de 70 veces más rápido que al hacerlo sobre una CPU multinúcleo, lo que significaba que un modelo de Deep Learning pase varias semanas entrenándose a solamente un día.

¿Pero cuál es la razón de esto? El por que es que cuando se entrena una red neuronal la cual esta compuesta por muchas capas con diferentes neuronas cada una de las neuronas realiza un calculo que es independiente de las demás, este calculo no solo es independiente, si no que es el mismo calculo que se ejecuta en

el resto de las neuronas que se encuentran dentro de la misma capa, y esta es una de las características que nos permite distribuir estas tareas en los diferentes hilos de procesamiento que tiene la GPU, pero no solo es eso, otra característica que es paralelizada es el procesamiento de los datos de entrada los cuales pueden ser procesados de manera independiente, aunado a esto, los fabricantes de procesadores están diseñando arquitecturas con un nivel mayor de especialización para hacer frente a la demanda del uso de especializado de este tipo de procesadores, un claro ejemplo de ello son los **Tensor Cores** los cuales son núcleos de procesamiento que se han de encargar de resolver una única tarea, esta tarea consta de multiplicar dos matrices de tamaño 4x4 y agregarla a otra matriz, y puede que no parezca muy impresionante pero esta operación ahorra la tarea de paralelizar la tan demandada multiplicación de matrices y adicional a esto es que la operación se ejecuta en un único ciclo de reloj, y probablemente las matrices que se han de procesar sean mucho mayores a 4x4, pero los Tensor Cores nos permiten subdividir una matriz mas grande en bloques mas pequeños y de esta forma calcular el subproducto de cada uno de estos bloques.

Conclusión

Actualmente el uso de este tipo de tecnología se encuentra un poco mas al alcance de alguien que quiera solucionar un problema haciendo uso inteligencia artificial, y es que existen varias herramientas, tanto hardware como software e información, que facilitan este tipo de tareas, del lado del software existe TensorFlow, PyTorch o Fast.IA los cuales nos permiten sacar partida de la aceleración por hardware de la GPU.