Singular Value Decomposition on GPU using CUDA

*by Sheetal Lahabar, P J Narayanan*

Los GPU modernos son utilizados para muchas tareas generales y han sido aclamados por sus co-procesadores relativamente baratos y altamente eficientes dado su gran poder computacional. En el paper de S. Lahabar, P. J. Narayanan llamado “Singular value decomposition on GPU using CUDA” ellos buscan presentar la implementación de la Descomposición de Valores Singulares de una matriz densa en GPU utilizando el modelo de programación CUDA. A continuación se sintetizan lo expuesto en cada sección.

**Descomposición de Valores Singulares (DVS):**

La DVS es una importante técnica para la factorización de una matriz rectangular y real. Se utiliza para computar una matriz pseudoinversa, resolver ecuaciones lineales, resolver un problema de minimización de errores cuadŕaticos y para encontrar una matriz de aproximación. También se usa comúnmente en aplicaciones relacionadas al análisis de Componentes Principales, procesamiento de señales e imágenes, reconocimiento de patrones, etc.

La DVS de una matriz m x n A es la factorización de la forma:

A = U ΣVT

Donde U es una matriz ortogonal de m x m, V de n x n y Σ es una matriz diagonal de m x n.

**Trabajos relacionados:**

Han habido varios esfuerzos para paralelizar el algoritmo de DVS en arquitecturas como FPGA, GPU, Cell Processores, etc.

**Algoritmo DSV:**

El algoritmo explota el parallelismo de la arquitectura del GPU y logra alto rendimiento computacional. La biodiagonalización de la matriz es realizada completamente en el GPU utilizando la optimizada librería CUBLAS. El paper utiliza una implementación híbrida para la diagonalización de la matriz que divide las computaciones entre el CPU y el GPU, dando buenos resultados en el desempeño.

El algoritmo consta de 4 pasos principalmente=

1: B ← Q T AP {Bidiagonalización de A a B}

2: Σ ← X T BY {Diagonalización of B to Σ}

3: U ← QX

4: V T ← (P Y ) T {Computar matrices ortogonales U y V T and DSV of A = U ΣV T }

Cada paso puede ser realizado por funciones CUDA BLAS. Este realiza funciones altamente eficientes para operaciones de matriz-vector, matriz-matriz y calculo de normas. El rendimiento de las librerias del GPU dependen de la distribución de los datos y el uso que le des a la librería. Experimentos demuestran que CUBLAS funciona de mejor manera con matrices de dimensiones de múltiples de 32 dado problemas con la alocación de la memoria.

Hay varias librerías numéricas como ATLAS Intel Math Kernel que son usadas ampliamente para diferentes aplicaciones en el CPU. Están diseñadas para lograr alta precisión así como gran memoria bandwidth. Intel MKL LAPACK da mejor rendimiento que el estándar LAPACK en procesadores Intel.

**Resultados:**

En esta sección analizan el rendimiento del algoritmo con la implementación optimizada del CPU de DVS en MATLAB e Intel MKL 10.0.4 LAPACK. Para esto, se generaron 10 matrices densas de manera aleatoria. El algoritmo DVS se ejecutó 10 veces para cada matriz.

Se logró computar el DVS de matrices muy grandes de orden hasta 14 K, lo cual es imposible realizar en un CPU dado la limitación de la memoria. Los GPU’s estan limitados a números del sistema del punto flotante aunque eso está cambiando en las nuevas generaciones. El error de precisión fue de menos de 0.001% en las matrices aleatorias con las que experimentaron. El enfoque desarrollado en CUDA y otras librerías disponibles pueden servir para resolver tareas gráficas y no gráfincas.