Introducción al Cálculo Numérico en Procesadores Gráficos

Docentes

- Kolton, Alejandro B.; Instituto Balseiro; UNCuyo y CONICET.

Curso

El curso propuesto tiene como objetivo principal facilitar a las y los estudiantes el aprendizaje de técnicas de cálculo y procesamiento de datos usando aceleradores gráficos, en el marco del paradigma de hardware y software masivamente paralelo.

El curso tiene un carácter fundamentalmente práctico, de laboratorio de computación, y está orientado a la aceleración del cálculo numérico típico y básico que se encuentra en ciencias e ingenierías. En cada clase, luego de una breve exposición de los conceptos, herramientas y vínculos a la información más relevante, se continúa con la discusión y aplicación de los mismos en ejemplos simples pero concretos, para luego dar lugar a la experimentación usando un cluster de cálculo con procesadores gráficos.

Requisitos

Son requisitos para el curso, tener un conocimiento básico de lenguaje C y de programación en general en cualquier lenguaje. Es necesario disponer de una computadora personal, con cualquier sistema operativo, para poder acceder remotamente a los servidores durante la clase. El clúster estará también disponible fuera del horario de clase para que puedan practicar y continuar investigando por cuenta propia. No es necesario tener una computadora personal con GPU. Tener en cuenta que para conectarse al clúster de GPUs desde fuera de la red del CAB será necesario poseer o tramitar personalmente un acceso vía VPN a quien corresponda.

Cronograma

El curso tendrá una carga horaria total de 64 horas (media materia), repartidas en 16 encuentros de 4 horas, 2 veces a la semana.

Equipamiento

Durante el curso, alumnas y alumnos tendrán cuentas en un cluster con GPUs modernas donde podrán practicar las 24hs vía acceso remoto seguro. El mismo tendrá instaladas versiones recientes de NVIDIA CUDA toolkit, NVIDIA HPC-SDK, CUPY, RAPIDS AI y compiladores de SYCL.

Programa analítico

1. Modelos de programación en paralelo. Paralelismo masivo en GPUs. Paralelismo en la arquitectura CUDA, ecosistema CUDA. Cluster de GPUs.

2. Herramientas para programar GPUs (lenguajes, compiladores, profilers, entornos). Ejemplos sencillos de paralelización y programación de GPUs usando CUDA C/C++/Fortran/Python/Julia y openACC.

3. Ejemplos de uso de bibliotecas para facilitar el cálculo numérico paralelo: patrones paralelos genéricos (THRUST), álgebra lineal densa y rala (CUBLAS, CUSPARSE, CUSOLVER), transformadas rápidas de Fourier (CUFFT), generadores de números aleatorios (CURAND y otras bibliotecas). Ejemplos equivalentes para Python vía CUPY.

4. Ejemplos de aplicación en cálculo numérico a problemas de Ciencias o Ingeniería: ecuaciones diferenciales ordinarias y parciales, dinámica estocástica (Langevin y Montecarlo), dinámica molecular. Breve introducción a GPGPU en la nube (Google Collaboratory, Saturn Cloud) y a ciencias de datos en GPU con RAPIDS AI.

5. Optimización básica y buenas prácticas de programación en CUDA.

6. Breve introducción a SYCL como alternativa a CUDA.

Evaluación:

Para aprobar es necesario haber asistido al 80% de las clases así como desarrollar y presentar un proyecto propio que aplique las técnicas aprendidas en un problema concreto de cualquier especialidad. Aunque ya disponga de una computadora personal con una GPU y toolkit adecuados para el curso, los códigos con las resoluciones de los ejercicios deberán estar adaptados para compilar y correr en el cluster. La nota final depende del proyecto final y su exposición y del grado de participación en las clases.

Bibliografía:

- NVIDIA Inc., CUDA Toolkit Reference Manual, última versión. https://docs.nvidia.com/cuda/
- David B. Kirk, Wen-mei W. Hwu. Programming Massively Parallel Processors: A Hands-on Approach, Morgan Kaufmann, 2010.
- Jason Sanders, Edward Kandrot, CUDA by Example, Addison-Wesley, 2010.
- Robert Farber, CUDA Application Design and Development, Morgan Kaufman, 2011
- Shane Cook, CUDA Programming, Morgan Kaufman, 2012.
- Duane Storti, Mete Yurtoglu, CUDA for Engineers: An Introduction to High-Performance Parallel Computing.
- Gregory Ruetsch and Massimiliano Fatica, CUDA Fortran for Scientists and Engineers. Andrzej Chrzęszczyk, Jacob Anders, Matrix computations on the GPU: CUBLAS, CUSOLVER and MAGMA by example, Version 2017.
- Nicholas Wilt, The CUDA Handbook: A Comprehensive Guide to GPU Programming.
- Tolga Soyata, GPU Parallel Program Development Using CUDA.
- James Reinders et al, Data Parallel C++: Mastering DPC++ for Programming of Heterogeneous Systems using C++ and SYCL.
- Sitios web: stackoverflow, parallel for all blogs, github.