Easy, Effective, Efficient: GPU Programming in Python with PyOpenCL and PyCUDA

ANDREAS KLÖCKNER
Courant Institute of Mathematical Sciences, New York University, USA

Why this workshop?

Throughput-oriented GPU-like architectures are firmly established as a way to gain much performance over latency-oriented CPU-like architectures for a significant number of computational problems. Programming these machines is–and will be, for the foreseeable future–fraught with some extra complexity compared to sequential programs. This workshop discusses one approach to managing this complexity.

Scripting languages are in many ways polar opposites to GPUs. GPUs are highly parallel, subject to hardware subtleties, and designed for maximum throughput. On the other hand, scripting languages (such as Python) favor ease of use over computational speed and do not generally emphasize parallelism. PyOpenCL is a package that joins the two together. This course aims to show you that by combining these opposites, a programming environment is created that is greater than just the sum of its two parts. One key to increasing programmer productivity in this setting is the easy application of run-time code generation (RTCG). The course will cover this and other techniques in detail.

PyOpenCL can be used in a large number of roles, for example as a prototyping and exploration tool, to help with optimization, as a bridge to the GPU for existing legacy codes (in Fortran, C, or other languages), or, perhaps most excitingly, to support an unconventional *hybrid way of writing high-performance codes*, in which a high-level controller generates and supervises the execution of low-level (but high-performance) computation tasks to be carried out on varied GPU- or GPU-based computational infrastructure. You will learn about each of these roles and how to get the most out of PyOpenCL (and also PyCUDA). In doing so, you will also gain familiarity with the OpenCL computation interface as well as a number of GPU (and CPU) hardware architectures.

Sponsor: ITMAN and DCAMM is sponsoring some refreshments for participants of the workshop during the day.

Deadline for signing up per E-mail is August 10, 2011 and is free of charge. There is limited capacity in the Lab so sign up early is recommended.

About the workshop

The workshop will take place in three sessions, each about an hour long, with practical lab sessions in between.

Introduction

- Intro to OpenCL
- (Brief) Intro to Python, numpy (if necessary)
- The OpenCL Universe: CL on CPUs and various GPUs
- PyOpenCL: OpenCL Scripting Mechanics for GPU and multi-core

Code Generation, Hybrid codes

- Motivation for Hybrid Codes: GPUs and Scripting
- GPU arrays, custom element-wise and reductive operations
- Run time code generation: How and Why
- Advanced code generation: Templating
- Heuristics and search patterns for automated tuning, performance measurement

Perspectives, Applications

- PyOpenCL and mpi4py
- Hybrid Development: Interfacing Python with Fortran and C/C++
- A brief look at PyCUDA
- An example: Gas dynamics and EM on complex geometries

Signing up for the workshop is required and is free of charge.

Prerequisites: This course will offer a brief introduction to GPU architecture and programming. Previous exposure to the topic will be helpful.

Registration: Register for the workshop by E-mail to Assoc. Prof. Allan P. Engsig-Karup (apek@imm.dtu.dk), Section for Scientific Computing, Department of Informatics and Mathematical Modelling.

August 17, 2011 · 9:00-17:00 · Building 305 R017 (VR-Lab)

Detailed Program (Tentative)

9:00 – 11:00	Introductory Session • Python, numpy, GPUs, OpenCL • Basic PyOpenCL • Contexts, Buffers, Events and such: A tour of the PyOpenCL runtime • More advanced PyOpenCL usage • The OpenCL device language • PyOpenCL: What comes in the box • Notes on CL implementations
11:00 – 11:10	*Break*
11:10 – 11:45	Lab Session I (intro) • Python, numpy recap • Simple PyOpenCL practice • Elementary Benchmarking • Microbenchmarking
11:45 – 13:00	*Lunch* (If you would like to maximize lab time, we suggest bringing lunch. There is an area with tables and chairs near the lab. Andreas will be available to help in the lab during lunch hours.)
13:00 – 13:30	Lab Session I cont'd
13:30 – 15:30	Advanced Session • The mechanics behind PyOpenCL • Run time code generation: How and Why, Templating • Automated tuning • mpi4py and PyOpenCL • Hybrid Development: Interfacing Python with Fortran and C/C++ • A brief look at PyCUDA • An example: Gas dynamics and EM on complex geometries
15:30 – 15:40	*Break*
15:40 – 18:00	Lab Session II (advanced) • TBD