gpyfft Documentation

Release 0.1

Gregor Thalhammer

June 06, 2012

CONTENTS

CHAPTER

ONE

GPYFFT

A Python wrapper for the OpenCL FFT library APPML/clAmdFft from AMD

1.1 Introduction

AMD has created a nice FFT library for use with their OpenCL implementation called AMD Accelerated Parallel Processing Math Libraries This C library is available as precompiled binaries for Windows and Linux platforms. It is optimized for AMD GPUs. (Note: This library is not limited to work only with hardware from AMD, but according to this forum entry it currently yields wrong results on NVidia GPUs.)

This python wrapper is designed to tightly integrate with pyopencl. It consists of a low-level cython based wrapper with an interface similar to the underlying C library. On top of that it offers a high-level interface designed to work on data contained in instances of pyopencl.array.Array, a numpy work-alike array class. The high-level interface is similar to that of pyFFTW, a python wrapper for the FFTW library.

Compared to pyfft, a python implementation of Apple's FFT library, AMD's FFT library offers some additional features such as transform sizes that are powers of 2,3 and 5, and real-to-complex transforms. And on AMD hardware a better performance can be expected, e.g., gpyfft: 280 Gflops compared to pyfft: 63 GFlops (for single precision, accurate math, inplace, transform size 1024x1024, batch size 4, on AMD Cayman, HD6950).

1.2 Status

This wrapper is currently under development.

1.2.1 work done

- low level wrapper (mostly) completed
- high level wrapper: complex (single precision), interleaved data, in and out of place (some tests and benchmarking available)
- creation of pyopencl Events for synchronization

1.2.2 missing features

- debug mode to output generated kernels
- documentation for low level wrapper (instead refer to library doc)

- define API for high level interface
- high level interface: double precision data, planar data, real<->complex transforms
- high level interface: tests for non-contiguous data
- handling of batched transforms in the general case, e.g. shape (4,5,6), axes = (1,), i.e., more than one axes where no transform is performed. (not always possible with single call for arbitrary strides, need to figure out when possible)

1.3 Requirements

- python
- pyopencl (git version newer than 4 Jun 2012)
- cython
- APPML clAmdFft 1.8
- AMD APP SDK

1.4 Installation

1. Install the AMD library:
   - install clAmdFft
   - add clAmdFft/binXX to PATH, or copy clAmdFft.Runtime.dll to package directory
   - edit setup.py to point to clAmdFft and AMD APP directories

Then, either:

2. python setup.py install

Or:

3. inplace build: python setup.py build_ext –inplace

1.5 License:

LGPL

1.6 Tested Platforms

OS	Python	AMD APP	OpenCL	Device	Status
Win7 (64bit)	2.7, 64bit	2.7	OpenCL 1.2, Catalyst 12.4	AMD Cayman (6950)	works!
Win7 (64bit)	2.7, 32bit	2.7	OpenCL 1.1 AMD-APP-SDK-v2.4 (595.10)	Intel i7	works!
Win7 (64bit)	2.7, 32bit	2.7	OpenCL 1.1 (Intel)	Intel i7	works!
Win7 (64bit)	2.7, 32bit	2.7	OpenCL 1.0 CUDA 4.0.1 (NVIDIA)	Quadro 2000M	Fails
Win7 (64bit)	2.7, 32bit	2.7	OpenCL 1.2 AMD-APP (923.1)	Tahiti (7970)	works!
Win7 (64bit)	2.7, 32bit	2.7	OpenCL 1.2 AMD-APP (923.1)	AMD Phenom IIx4	works!

3. Gregor Thalhammer 2012

CHAPTER

TWO

INDICES AND TABLES

- *genindex*
- *modindex*
- *search*