anuga-cuda - the GPU based ANUGA

Documentation

Documentation is under doc directory, and the html version generated by **Sphinx** under the directory *doc/sphinx/build/html/*

Install Guide

1. Original ANUGA is required
2. For CUDA version, PyCUDA is required
3. For OpenHMPP version, current implementation is based on the CAPS OpenHMPP Compiler
    - When compiling the code with **Makefile**, the NVIDIA device architecture and compute capability need to be specified
        - For example, GTX480 with 2.0 compute capability
          `HMPP_FLAGS13 = -e --nvcc-options -Xptxas=-v,-arch=sm_20 -c --force`
        - For GTX680 with 3.0 compute capability
          `HMPP_FLAGS13 = -e --nvcc-options -Xptxas=-v,-arch=sm_30 -c --force`
    - Also the path for **python**, **numpy**, and **ANUGA/utilities** packages need to be specified
    - Defining the macro **USING_MIRROR_DATA** in **hmpp_fun.h**
      `#define USING_MIRROR_DATA`
      will enable the advanced version, which uses OpenHMPP Mirrored Data technology so that data transmission costs can be effectively cut down, otherwise basic version is enabled.

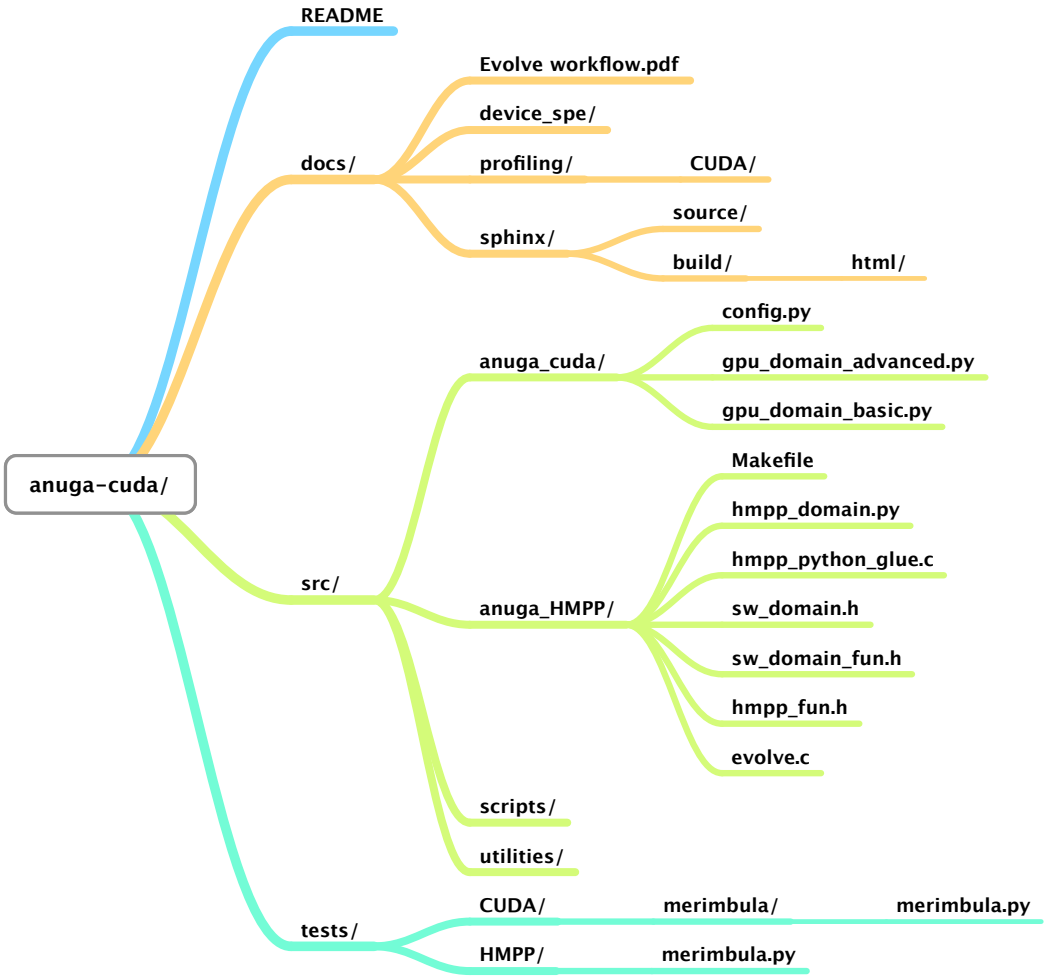
Environment Vars

Please add following vars to you **.bashrc** or **.bash_profile** file

```
export ANUGA_CUDA=/where_the_anuga-cuda/src

export $PYTHONPATH=$PYTHONPATH:$ANUGA_CUDA
```

Basic Code Structure



- **README** (What you are reading)
- **docs/** Documentation directory
  - **codeStructure.pdf** The diagram above
  - **Evolve workflow.pdf** The overall workflow of the evolve procedure. This includes all the function dependency and function interfaces, which is helpful to understand the evolve procedure of ANUGA
  - **device_spe/** Some device specifications of our working station
  - **profiling/** Profiling results
    - **CUDA/** All the profiling results on CUDA implementation
  - **sphinx/** The Sphinx generated documentation
    - **source/** The source files for Sphinx based documents
    - **build/** The generated documents
      - **html/** HTML version documentation
- **src/** Source code directory
  - **anuga_cuda/** The CUDA implementation

        - **config.py** Detail configuration for the CUDA implementation, including the path for all the kernel functions, optimal CUDA thread block configuration, etc.
        - **gpu_domain_advanced.py** Python Class for CUDA implementation in advanced version
        - **gpu_domain_basic.py** Python Class for CUDA implementation in basic version
    - **anuga_HMPP/** The OpenHMPP implementation
        - **Makefile** The Makefile
        - **hmpp_dimain.py** Python Class for OpenHMPP implementation
        - **hmpp_python_glue.c** The Python/C API to set up communication between Python ANUGA and OpenHMPP.
        - **sw_domain.h** The C Struct type **domain** used in C implementation to access mesh information generated in Python ANUGA
        - **sw_domain_fun.h** Connect C Struct type **domain** to all mesh information
        - **hmpp_fun.h** All function declarations.
        - **evolve.c** The evolve procedure
    - **scripts/** Some useful bash script
    - **utilities/** Utilities for sorting mesh information, checking results, etc.
- **test/** Testing cases
    - **CUDA/** Testing cases for CUDA implementation
        - **merimbula/** Merimbula testing case directory
            - **merimbula.py** Merimbula testing case
    - **OpenHMPP/** Testing cases for OpenHMPP implementation
        - **merimbula.py** Merimbula testing case

Examples

Running Merimbula model with CUDA implementation.

```
$ python merimbula.py -gpu
```

With pair-testing.

```
$ python merimbula.py -gpu -test
```

With rearranged mesh information.

```
$ python merimbula.py -gpu -rg
```

Author

Mail to Zhe Weng (John)