DLI Accelerated Data Science Teaching Kit

Module 1 Lab:

Introduction to Parallel Programming

The purpose of this lab is to introduce the student to the CUDA platform along with their capabilities. The student is not expected to understand the details of the code, but should understand the process of compiling and running code. The lab consists of the following four notebooks. Each notebook is individual and you will run the notebooks on the Jetson Nano device.

- Notebook1: Introduction to CUDA and PyCUDA
- Notebook2: Introduction to CUDA Python with Numba
- Notebook3: Array summation and matrix multiplication
- Notebook4: Benchmarking speed: NumPy, CuPy

To run the notebooks on your Jetson, you will use l4t-ml container which already has the prerequisite libraries. It's a machine learning container that contains TensorFlow, PyTorch, JupyterLab, and other popular ML and data science frameworks such as scikit-learn, scipy, and Pandas pre-installed in a Python 3.6 environment.

Run the container

First, you should pull one of the `l4t-ml` container tags from the list, corresponding to the version of JetPack-L4T that you have installed on your Jetson. For example, if you are running the latest JetPack 4.6 (L4T R32.6.1) release:

```
sudo docker pull nvcr.io/nvidia/l4t-ml:r32.6.1-py3
```

Then to start an interactive session in the container, run the following command:

```
sudo docker run -it --rm --runtime nvidia --network host
nvcr.io/nvidia/l4t-ml:r32.6.1-py3
```

You should then be able to start a Python3 interpreter and `import` the packages above.

Please check out the link for more information.

Summary of exercises

Notebook1: Introduction to CUDA and PyCUDA

- Doubling the value of elements in an array
    - Step1: Getting started

    - Step2: Transferring data to the GPU
    - Step3: Executing a Kernel
    - Abstracting Away the Complications
    - Shortcuts for Explicit Memory Copies
- Addition of two 1D arrays
- Addition of matrices
- Multiplication of matrices
- Linear combination of variables
- Numba and PyCUDA

Notebook2: Introduction to CUDA Python with Numba

- Compiling Python code with @jit
    - Compiling functions on the CPU
- Compilation options
- Type inference
- Creating Numpy universal functions
    - Universal functions
    - Generalized ufuncs on the GPU
- Writing Device Functions

Notebook3: Array summation and matrix multiplication

- Array summation: Traditional computing
- Array summation: Parallel computing
    - Compiling
    - Memory transfer
    - Execute the kernel
- Matrix multiplication: Traditional computing
- Matrix multiplication: Parallel computing (Striding)
- Matrix multiplication: Parallel computing (Shared memory)

Notebook4: Benchmarking speed: NumPy and CuPy

- Creating an array with NumPy
- Creating and array with CuPy
- Trigonometric function

- Multiplying the array
- Performing multiple operations on the array