Nvidia Jetson Remote Runners

Background:

The HPSS lab, run by Dr. Mehmet Belviranli on campus, utilizes a variety of embedded hardware systems called the Nvidia Jetson series. These devices are used since we focus on heterogeneous computing architectures such as GPUs, FPGAs, and AI accelerators. These devices are utilized for teaching courses (such as the beyond CPU computing class) to performing research/preparing publications. However, each device has its own distinct hardware and computing capabilities. Managing such capabilities is difficult and significant work needs to be done for manually managing the correct software versioning on each device. Automatically handling high-level applications can significantly reduce manual effort.

Project Description:

This project will focus on the Nvidia Jetson platform family of devices. These devices are system-on-ship (SoC) development boards which we utilize daily. However, as it exists despite these devices being from the same family of devices they must all be utilized separately.

High Level Overview:

1. Establish connect to each Jetson device for a “host” (calling device)
2. Transfer source code and data files to Jetson device
3. Compile source (if applicable) for Jetson device and prepare binary
4. Run application with input data or files and collect stdout/stderr results to files
5. Transfer stdout/stderr back to host and aggregate into an output format

Project Goals:

1. Develop an open source tool for executing applications on Nvidia Jetson platforms
2. Design and implement an interface for running applications remotely on the Jetson devices
3. Design an input format for the applications based on stdout and/or input file needs
4. Design an output format for easily reading final results, including packaging output files if they exist
5. Design and implement the data movement schemes on a host machine
6. Full documentation of functionality for usability and of the code itself such that future work can take place

Additional/Stretch Goals:

1. Utilize Python virtual environments to run applications
2. Utilize and learn Docker for running all applications
3. Implement virtual environments and Docker images for running applications
4. Distribution setup via a package manager (such as PyPi) or a git clone and build methodology

Desired Skill Set:

- Python and/or C++
- GCC (or other similar compiler tools)

- Familiarity with ssh and scp could be beneficial but not required

Preferred Team Size: 3-5 Students

Location of Work: Mines Campus

Client Liaison:
Justin Davis
jcdavis@mines.edu