CmpE 275 Project 2 Technologies

Project 2:

Blended or hybrid applications that employ interpreted languages (python, Java) that utilize compile assets are becoming more common place as the need for compute intensive operations to analyze data. Furthermore, the affordability of adding High Performance Computing (HPC) resources to a enterprise design has over the last five years become common with turn-key solutions that are delivered racked, configured, and pre-installed with software stacks.

Project 2 provides an opportunity to look at some of these approaches and challenges faced in HPC approach to distributed computing. Because of the limited time available, the project has been scaled back to:

1. Paper with optional coding and timing
2. Lightning talk (ref. https://en.wikipedia.org/wiki/Lightning_talk) to be presented as the final exam.
3. **Teams must reform to sizes of 2 or 3 people.**

Suggested list of technologies

Please install and verify the following technologies for project 2. Note I would like to to install the latest set of packages compiled onto your computer. While package installers like yum, apt-get, RPMs, or pre-bundled binaries – you should download the source code as you may need to debug or optimize the build. Lastly, there maybe additional packages that you may need to build prior to installation as well as for your team's needs. I will help you to install gcc 6.2 however, the remaining packages are your responsibility.

Likely Technologies:

- Gnu gcc/g++ version 6.2 or Clang 4.0 (or newer)
- Boost (latest version, optional)
- Cmake
- OpenMPI
- Swig
- Python (likely already installed)
- Python Flask (optional)

Installation strategies and hints:

- Installing into the default location (/usr) may require root privilege. **It is highly advisable that you do not install compilers into /usr as it may interfere with the default (if any).** Furthermore, you may want multiple compiler and versions on your system.

  - To help with your installation, create a base directory in /opt or /usr/local to not pollute your system directories/files.
  - For instance, /usr/local/tools, would have a typical configure of:

```
./configure --prefix=/usr/loal/tools
```

- Read pre-requisites closely as a package will have dependencies. If you install into a base directory you may need to declare where this is for each package.