Data Processing

Computing architecture

L. Kos[1] G. Simič[1]

[1]Faculty of Mechanical Engineering
University of Ljubljana

EUDAT-PRACE summer school, September 2019

Overview

Demonstration and hands-on login.galileo.cineca.it

- Environment modules
- Conda environment
- Python on HPC Systems
- Using batch system

Environment modules for HPC

- All additional software on the HPC system is made available through the environment module tool. It provides a way to rationalize software and its environment variables allowing custom build software versions for HPC.
- TCL or LUA based module environment system is provided on many Linux systems.
- Commonly used Easybuild and Spack HPC building system provide modules for the software build.

The same environment setup is used in different unix shells. Commonly, the following shell variables are being set:

PATH Search path for commands provided by module outside from standard `/usr/bin`

LD_LIBRARY_PATH Shared libraries search path if not linked with rpath.

MANPATH, LIBPATH, CPATH, PKG_CONFIG_PATH Search paths for manual, compiler linker, C includes and package config, respectively.

custom Specific environment variables for use by the module or other modules.

Single command module handles setting environment variables for the shell.

```
[a08trb24@r033c01s05 ~]$ module
Modules Release Tcl 3.1.6 ($RCSfile: modulecmd.tcl,v $ $Revision: 1.112 $)
        Copyright GNU GPL v2 1991
Usage: module [ switches ] [ command ]
Switches:
        -t              terse format avail and list
        -l              long format avail and list
Commands:
        list            |   add|load             modulefile [modulefile ...]
        purge           |   rm|unload            modulefile [modulefile ...]
        reload          |   switch|swap          [oldmodulefile] newmodulefile
                        |   display|show         modulefile [modulefile ...]
                        |   avail                [modulefile [modulefile ...]]
                        |   whatis               [modulefile [modulefile ...]]
                        |   help                 [modulefile [modulefile ...]]
                        |   path                 modulefile
                        |   paths                modulefile
                        |   use                  dir [dir ...]
                        |   unuse                dir [dir ...]
                        |   source               scriptfile
                        |   apropos|keyword      string
                        |
        initlist        |   initadd              modulefile
        initclear       |   initprepend          modulefile
                        |   initrm               modulefile
```

Modules are system dependent and are on CINECA HPC systems divided in several profiles (recommended software groups):

profile/base (default/stable) and tested compilers, libraries, tools

profile/advanced libraries and tools compiled with different setups than the default

profile/chem (phys, bioinf, astro, ...) "domain" profiles with the application softwares specific for each research field

profile/archive old or outdated versions of software still available but not recommended to use anymore.

There is also `modmap -all` command provided at CINECA clusters only to list the environment modules available inside all profiles. Each profile contains compilers, libraries, tools, and applications.

Conda is a package manager for users (on HPC). To get the package manager use Miniconda installation page for selected platform `https://docs.conda.io/en/latest/miniconda.html` For 64-bit x86 Linux (Most HPCs) use:

```
wget https://repo.anaconda.com/miniconda/
Miniconda3-latest-Linux-x86_64.sh
bash Miniconda3-latest-Linux-x86_64.sh \
 -b -p ${HOME}/miniconda3
source ${HOME}/miniconda3/etc/profile.d/conda.sh
conda create -f conda-data_analysis-env.yaml
conda activate data_analysis
```

If Anaconda is available on the system, then conda can be used from there:

```
# Load anaconda module
module load anaconda/2019.07
# Initialize conda shell
conda init bash
# Create custom environment
conda create --prefix my_own_env
# Activate environment
conda activate my_own_env/
# Install packages
conda install netCDF4 numpy scipy basemap
conda install matplotlib jupyter xlrd pandas
```

Running batch scripts

```
# Get info about node status
squeue

# Script example.sbatch
#!/bin/bash
echo "Hello world"

# Make script executable
chmod +x example.sbatch
# Run script
srun example.sbatch
```

Please join `https://www.futurelearn.com/courses/python-in-hpc`

```
git clone https://github.com/csc-training/hpc-
    ↪ python.git
module load autoload mpi4py
```

Selected demos can be run directly by setting alias

```
prun='srun --nodes=1 --ntasks-per-node=12 --time
    ↪ =10:00 --account=train_ceudat19 --partition=
    ↪ gll_usr_prod'
prun python mpi-hello.py
```