Cluster module of the week

An introduction to cluster usage

April 5, 2019

Overview

Short Motivation

Does everybody have access to Blaustein? If not:

- Find someone with an account who is willing to share it with you
- Add your public ssh key on blaustein into .ssh/authorized_keys

Short Motivation

What do I need to run a job on Blaustein

- Access -> ssh
- The right programs -> modules
- Some way to tell the cluster what to do -> Jobfiles

Blaustein Hardware

- 32 compute Nodes
  - 2 x Intel Xeon, Haswell architecture
  - 2 x 12 Cores with 2 x 24 threads (Hyperthreading)
  - 128 GB DDR 4 Ram

ssh

ssh into a JURECA: ssh pronold1@jureca08.fz-juelich.de -AX

- -A : Agent forwarding (use your local ssh key, get access to your github repos)
- -X : Enables X11 forwarding (for using graphics)

sshfs

Mount ssh folders locally

- mount : sshfs -o idmap=user pronold1@jureca.fz-juelich.de:/p/project/cjinb33/jinb3330 $HOME/remote
- unmount : fusermount -u $HOME/remote

Maybe add those commands to your .bash_aliases
alias sshfs-jureca="sshfs -o idmap=user pronold1@jureca.fz-juelich.de:/p/project/cjinb33/jinb3330 $HOME/remote"

Modules

What are Modules?

- Modules are a way of managing the user environment
- Modules let users modify the environment on the fly
- Modules provide preinstalled software
- Modules can be loaded

Modules

Basic module commands

- module -shows the list of module commands
- module avail -shows a list of "available" modules
- module list -shows a list of loaded modules
- module load $name -loads a module
- module unload $name -unloads a module
- module help $name -prints help for a module
- module whatis $name -prints info about the module
- module purge -unload all modules
- module swap $name1 $name2 -swap two module

SLURM

What is SLURM

- SLURM is a batch scheduling software
- Simple Linux Utility for Resource Management
- allocating exclusive and/or non-exclusive access to resources (computer nodes) to users for some duration of time so they can perform work
- providing a framework for starting, executing, and monitoring work (typically a parallel job such as MPI) on a set of allocated nodes

SLURM

Basic SLURM commands (On the command line)

- squeue - shows all jobs in the queue
- squeue -u $username - shows only your jobs
- sinfo - shows partition/node state
- scancel $jobid - cancels a job (find the jobidwith squeue!)

In the following I will introduce the following concepts of starting jobs

- srun $jobfile - start a jobfile
- salloc - request resources for an interactive session
- sbatch $scriptname - launches a batch script

```
module load pystuff_new mpi/openmpi nest/2.10.0

srun -o /home/c.keup/jari/microcircuit/\%j.o \
-e /home/c.keup/jari/microcircuit/\%j.e \
--job-name microcircuit \
--mem=120G \
--time=12:00:00 \
--exclusive \
--ntasks 1 \
--cpus-per-task 48 \
--nodes 1 \
--mail-type=END,FAIL \
--mail-user=j.pronold@fz-juelich.de \
python Potjans_2014/example.py
```

SLURM

Running interactive jobs

- salloc -n 1 - request resources for an interactive job on one node
- srun $executable - Start job in interactive session
- Ctrl - D (or exit)- Quit interactive session

Interactive sessions can be quite handy for testing scripts on a compute node

```
salloc -n 1

module load pystuff_new mpi/openmpi nest/2.10.0

srun -o /home/c.keup/jari/microcircuit/\%j.o \
-e /home/c.keup/jari/microcircuit/\%j.e \
--job-name microcircuit \
--mem=120G \
--time=12:00:00 \
--exclusive \
--ntasks 1 \
--cpus-per-task 48 \
--nodes 1 \
--mail-type=END,FAIL \
--mail-user=j.pronold@fz-juelich.de \
python Potjans_2014/example.py
```

SLURM

SBTACH directives

- #SBATCH-t 1:00:00 -time of a job
- #SBATCH–nodes=2 -number of nodes
- #SBATCH–ntasks 2 -total number of slurmtasks requested (MPI Processes)
- #SBATCH–ntasks-per-node=1 -tasks per node
- #SBATCH –cpus-per-task=48 - Number of threads per task
- #SBATCH–mail-type=FAIL,BEGIN,END-send an email on events
- #SBATCH–mail-user=name@example.com -user email address
- #SBATCH-o filename -standard output file
- #SBATCH-e filename -standard input file
- #SBATCH –mem=10000 -reserve 10GB of memory per node

These directives have to be included in the batch file you write for your job

```
#!/bin/bash
#SBATCH -o /home/c.keup/jari/microcircuit/%j.o
#SBATCH -e /home/c.keup/jari/microcircuit/%j.e
#SBATCH --job-name microcircuit
#SBATCH --mem=120G
#SBATCH --time=12:00:00
#SBATCH --exclusive
#SBATCH --ntasks 1
#SBATCH --cpus-per-task 48
#SBATCH --nodes 1
#SBATCH --mail-type=END,FAIL # notifications for job done &
#SBATCH --mail-user=j.pronold@fz-juelich.de

module load pystuff_new mpi/openmpi nest/2.10.0
mpirun python Potjans_2014/example.py
```

Example: Running the Microcircuit

Assume all the Code from the slide before is saved in a file called microcircuit.jdf
The code can be submitted via
sbatch microcircuit.jdf

Screen Mulitplexer

Screen Multiplexer can be helpful when connected via ssh

- to have multiple terminals open
- to save current workspace
- to end ssh session while having jobs actively running
- to be sure to not lose current workspace even if ssh connection is lost for whatever reason (Change of internet connection from wifi to wired, loss of internet connection)

Focus on tmux

Tmux

How to use tmux

- tmux - opens a tmux session
- tmux new -s sessionName - opens new tmux session sessionName
- tmux attach -t sessionName - opens existing tmux session sessionName
- Ctrl - b + d : dettach from session
- Ctrl - b + % : opens vertical window
- Ctrl - b + " : opens horizontal window
- Ctrl - b + c : opens new window
- Ctrl - b + $windowNumber : opens window $windowNumber

Important: If you want to attach to a session that runs on cluster and you ssh into it, make sure to ssh to the correct headnode

Terminal Tricks

Some terminal movements that could be of advantage

- Ctrl - r: Search command history
- Ctrl - _: Undo last
- Ctrl - a: Jump beginning of line
- Ctrl - e: Jump end of line
- Add & to the end of the command: move program to the background
- Ctrl - k: Delete all text left from the cursor and save it to the clipboard
- Ctrl - k: Delete all text right from the cursor and save it to the clipboard
- Ctrl - y: Paste clipboard