eX3 workflows

Collecting best practices

moderated by Thomas Roehr

General constraints

- one project - multiple users
- heterogeneous target architectures (amd64 vs. arm64)
- shared resources in general
- popular and unpopular partitions / queues

Observation

- initial focus on ‘user’-centric ‘works-for-me’ approach
- initial permission setup (default umask) was not accounting for same user/group permissions
- docker - now disabled for majority of nodes due to security issues
- managing custom software stack via multiple / competing options:
  - module: https://curc.readthedocs.io/en/latest/compute/modules.html
  - conda: https://docs.conda.io/en/latest/
  - docker: https://docs.docker.com/engine/reference/builder/
  - spack: https://spack.io/
  - …

Module

- module avail <optional-pattern>
  - gives you a list of all installed modules
- module list
  - show loaded modules
- module show <name-of-module>
  - gives you the list of module specific environment setup
- module (un)load <name-of-module>

Modules (Custom)

Writing custom module files, e.g., in local_model:

1. Compile your component
2. Write your own module file as ‘custom-module’
3. Activate module
    - module use <user-folder>
        - append folder <user-folder> to load directories
    - module load custom-module

Observation

- filesystem usage
  - no dedicated backup of data - so every use might come with a different approach
    - syncing from Google Drive data bound to individual user accounts
  - no default folder structure, e.g. currently trying to seek a reasonable structure
- GPU support
  - nvidia-toolkit (will be preinstalled as module)
  - available, yet somewhat unclear what versions to use
  - starting point to monitor your GPU usage: nvidia-smi
    - https://developer.nvidia.com/nvidia-system-management-interface
- Running SLURM jobs with sbatch - good to know some details

SLURM - Workload Manager https://slurm.schedmd.com/overview.html

Using sbatch to send jobs (scripts) to the job queue:

- sbatch -A <user> -o <job.log> -J <jobname> -p <partition> –mincpus=<number-of-cpus> --time=0-00:10:00 myjob.sh

Options can also be encoded in the myjob.sh

```
#!/bin/bash
#SBATCH --job-name=myjob
#SBATCH --gres=gpu:1
```

For an interactive usage:

- srun -p <partition> --mincpus 128 --time 0-00:10:00 --pty bash

Manual allocation:

- salloc -N1 xterm
- depending on the usage of salloc you might have to cancel with: scancel <job-id>

SLURM - Workload Manager

Monitoring / Inspecting jobs:

- squeue --user=<user>
  - show all jobs of selected user
- scontrol show job <job-id>
  - list details of the particular job
- scontrol show reservations
  - Your job might conflict with current reservations so that you might have to adapt the --time option
  - when you own the reservation: sbatch --reservation=<reservation-id> ….
    - there are also ‘magnetic’ reservations where you will not need this

Potentially better …

- Keep user experience as close as possible to their single machine experience
  - start software as it is run on a single test machine
- Scripting (Bash/Python/…)
- Standard folder structure for projects
  - data
  - logs
  - software
    - scripts
    - venvs
    - ..
- Usage of README.md to point to relevant documentation

Links

- eX3: http://wiki.ex3.simula.no/doku.php?id=start
- SLURM:
  - https://slurm.schedmd.com/overview.html
  - https://curc.readthedocs.io/en/latest/running-jobs/slurm-commands.html
- Module: https://curc.readthedocs.io/en/latest/compute/modules.html
- Nvidia SMI: https://developer.nvidia.com/nvidia-system-management-interface