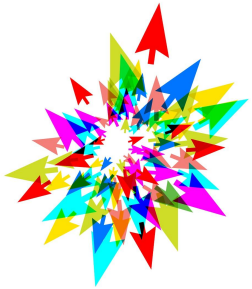
Intro to Compute Canada

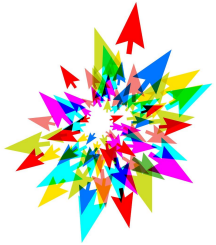
Evgeny Naumov



compute canada | calcul canada

Calcul Québec

Financial Partners



What is Compute Canada?

Overview

Calcul Québec

- Provides advanced research computing (ARC) resources to Canadian researchers
- Provides **support** to researchers using its infrastructure

Overview

Calcul Québec

- Operates in close cooperation with regional partners:
  - ACENET
  - Calcul Québec
  - Compute Ontario
  - WestGrid
- Funded by provincial, federal and university money

Common Environment

Calcul Québec

- Starting in 2017, there has been a push to standardize the software stack
- Goal is to present a common user interface across all assets
- This is the interface I will present, YMMV may vary on consortium servers

Bureaucracy

How to register?

Calcul Québec

- Register for an account (details by searching “compute canada account”)
  - At this point you can use limited RAS resources
- **Principal Investigators** (same eligibility as for federal grants) can get more resources for their groups through the RAC
  - PIs’ students can join their groups to access resources
  - Can be in more than one group - **choose the right account when running jobs!**

Glossary

- **RAS** - Rapid Access Service
  - 20% of compute; for small jobs; anyone with an account can use immediately
- **RRG** - Resources for Research Groups
  - 80% of compute; awarded on merit through RACs
- **RAC** - Resource Allocation Competition
  - Process by by which RRG compute is allocated

Consortium Accounts

Calcul Québec

- To use consortium accounts (e.g. on `calculquebec.ca` servers), need to register separately
- "*Apply for Consortium Account*" in the Compute Canada account page
- Separate credentials
- Computing environments may differ a bit
  - In the process of standardization

The Basics

Logging in

- SSH into the appropriate server using either your
  - Compute Canada credentials (.computecanada.ca)
  - Consortium creds (e.g. .calculquebec.ca)
- SSH best practises apply:
  - Private keys; PK encryption
- **The login node is not a compute node**
  - If you run calculations on it you will be ~~federally prosecuted~~ gently reminded not to!

Storage

Calcul Québec

- There are three fundamental storage areas
  - `/home/yourusername` (=: ~)
    - On remote server, NFS = slow!
    - 10GB of space = not good for data!
    - Use for config files, documents, code, etc.
  - ~/scratch/
    - Local to the server = fast!
    - 10 TB allocation
    - Use for… scratch data, rapid iteration
  - /project/…
    - RRG space, amount varies
    - Use for relevant project data

Running Jobs

Calcul Québec

- The login node is not a compute node!
- SLURM scheduler is used
- Several ways of using it
  - `sbatch` <your run script>
  - `srun` [flags] <your code>
  - `salloc` (for interactive session)
- Management commands
  - `squeue` check jobs
  - `scancel` cancel jobs
- Read the SLURM docs!

Running Jobs: Accounting

Calcul Québec

- Compute is metered!
- If you're in multiple projects:
  - Run the job under the **appropriate account**
    - `--account=` parameter to slurm
    - Your unix groups correspond to your accounts
  - Be careful with quota. Compute is limited, be efficient and considerate
- Conversely, don't use RAS (`def-xxx` account) if you have appropriate RRG (`rrg-xxx`). Slow!

There is much more to it

- Efficient use of HPC is hard
- Things to master:
  - When/how to use MPI
  - Efficient SLURM space and time requests (ask for enough, but no more)
    - `--mem`, `--n-tasks`, `--n-nodes`, `--time`, etc.
  - Checkpointing
  - Preemptible code
- It's worth taking some time to make sure you are not doing something inefficient!

GPU

Calcul Québec

- Not all servers have GPUs
  - `{cedar, graham}.computecanada.ca`
  - `helios.calculquebec.ca`
  - *Others, consortium sites have lists*
- Use `--gres=gpu:1` (2, 4, …) to request GPUs
- For distributed training:
  - Envvar **`SLURM_JOB_NODELIST`** gives node hostnames of allocated nodes
  - Use these to set up param servers, workers, etc.

Summary

- Register for account, join your PI's group
- Use SSH to log in
- Store code and config in `/home/yourname`
- Store big files in `~/scratch` or `/project/...`
- **NO BIG COMPUTE ON LOGIN NODES**
  - one-off tests/PoCs are fine
- Request GPU from SLURM with `--gres=gpu:1`

Software Installation

Primary Stacks

Calcul Québec

- **Lmod - `module`**
  - `module` command
  - Loads pre-configured software stacks
  - Handles libs, compatibility automagically
- **Easybuild - `eb`**
  - For building performance-sensitive scientific software not available through Lmod
- **Nix - `nix-env`**
  - For “personal” software like editors, stream processors
- **Pip - `pip`**
  - For python packages

Lmod

- The most important command
  - Selects compiler, interpreter, library versions
  - Automatically manages paths, etc
- Common commands:
  - `module load <name/version>` - load s/w
  - `module remove` <name>
  - `module av <name>` – quick search
  - `module spider <name>` – extensive search
- Implemented as bash functions, **if using other shell redefine them**
- Read the docs!

Easybuild

Calcul Québec

- An extensive, reproducible build system
- Usually used internally to create modules
- Can be used to build newer software versions
- Commands
  - `eb -S <name>` search for s/w
  - `eb <recipe>` build s/w
- Not needed often, but remember it's there!

Nix

Calcul Québec

- The package manager from nixos
- Rarely needed, nix packages already in default path
- Commands
  - `nix-env --install --attr <name>`
  - `nix-env --uninstall <name>`
  - `Nix-env --query --available <name>`
- CC has a fork of nixos nixpkgs:
  - https://github.com/ComputeCanada/nixpkgs
  - Feel free to clone and modify to your needs

Python

- **The golden rules:**
  - **AVOID CONDA**
    - Supported on a best-effort basis, discouraged
    - It infests your .bashrc leading to lots of stupid [PYTHON]PATH related tickets
  - **USE VIRTUALENVS**

```
$ module load python/x.y.x
$ mkdir venv
$ virtualenv venv
$ source venv/bin/activate
(venv) $ pip install foo
```

Python: wheels

Calcul Québec

- Wheels are precompiled python packages
- We have custom wheels for big packages
  - Compiled with fast libs and optimizations
  - *I'm working on a better build system to improve interoperability and keep it more up-to-date*
- Some important wheels:
  - `pip install ~~torch~~ torch_{cpu,gpu}==0.4.0`
  - `pip install ~~tensorflow~~ tensorflow-{cpu,gpu}`

Local Libs: `setrpaths.sh`

Calcul Québec

- Some software (e.g. MuJoCo) requires libraries to be installed locally.
- These have hardcoded **RPATH**s (shared object lookup paths), will die on CC environment
- The fix:
  - `setrpaths.sh --path <lib dir> [--add-origin]`
  - More info at $(search for “installing software local directory” on wiki)

Summary

1. Check modules (`module {av, spider}`)
2. Check EasyBuild (`eb -S/eb`)
3. Python:
   a. Conda = BAD
   b. Virtualenv = GOOD
4. Nix for minor software (your favourite editor or stream processor, etc.)

Support

How to get help?

- First, read the wiki! It's a fine document with concrete instructions:
  - https://docs.computecanada.ca
  - Consortia have own wikis (e.g. wiki.calculquebec.ca)
- Email support@computecanada.ca
  - That's me!
  - Tips:
    - Use Compute Canada account email address
    - Be specific
    - Give consent to look at your code (acct. option)

Things we resolve

- Admin/login/space/compute quota issues
- If pip-installed software crashes
  - Happens sometimes since our environment is nontrivial and we mix compilers
- If you need new/upgraded software
  - check easybuild first
- Access to ~~evil~~ proprietary software
- You're not sure how to implement something
  - Yes, we help you code your algos and run them on HPC
- Anything else you're stuck on!

Thank you and happy computing!