

Getting Started with Julia on HPC Clusters

Carsten Bauer

July 9 2024, JuliaCon Eindhoven

Julia Setup

Julia binaries: Keep it simple.

- Use the regular binaries
 - [juliaup](#) or [julialang.org](#)
- A system module might help you with packages
 - `module avail / module spider julia`
 - `module load julia`
- Generally, no need to compile from source.

Put the Julia depot on the parallel file system (PFS).

- PFS is often called “scratch”
 - High quotas
 - Writable (also from within compute jobs)
 - No backup of redundant data
- Set **JULIA_DEPOT_PATH** environment variable
- Watch out for **automatic deletion**
 - Workaround: touch files periodically

On heterogeneous clusters, use multiversioning.

- Nodes with different CPU kinds
 - e.g. login and compute nodes
 - re-triggering of package precompilation
- Set **JULIA_CPU_TARGET** environment variable
 - `export JULIA_CPU_TARGET="znver3;skylake,clone_all"`
 - `julia -C help`

Workflow:

Visual Studio Code

Challenges

- Running VS Code on cluster nodes
- Making the Julia extension work

Run VS Code on a cluster node via SSH

Login node

- Press **F1** and run the **Remote-SSH: Open SSH Host...** command.
 - `accountname@perlmutter.nersc.gov`

Compute node

- SSH ProxyJump
 - might not be possible at all
 - (requires full accounts at NERSC 🙄)

Use a Julia wrapper for the Julia extension

- [Julia: Executable Path](#) should point to a wrapper script. For Perlmutter:

```
#!/bin/bash

# Make julia available
module use /global/common/software/nersc/n9/julia/modules
module julia

# Pass on all arguments to julia
exec julia "${@}"
```

Run VS Code on a cluster node via “remote tunnels”

On the target node

- Download the **code** CLI and run
 - `code tunnel --verbose \`
`--cli-data-dir=$SCRATCH/.code_cli_data_dir`

Locally

- Press **F1** and run the **Remote Tunnels: Connect to Tunnel** command.

(also works with NERSC training accounts 😊)

Workflow: Jupyter

NERSC Jupyter

- <https://jupyter.nersc.gov/>