LCI Intermediate - User Software: Modules

Alan Chapman
Systems Analyst - Software Specialist
RTO Research Computing
Arizona State University

May 13th 2024

Module files

Let's dive into the fascinating world of "module files" on High-Performance Computing (HPC) systems, perfect for guiding new users on their quest to harness the incredible power of HPC.

Matrix of HPC: Module Files Unveiled

Imagine stepping into a vast library where every book provides you with specific tools and knowledge, but you can only carry a few at a time to solve a unique puzzle. In the realm of HPC, "module files" are your magical books. They are the secret sauce that makes working with diverse software environments as easy as flipping through a comic book. Let's explore the magical shelves of this library: Environment Modules, LMOD, and other sorceries.

Environment Modules: The Classic Saga

First off, we have Environment Modules, the original series that started it all. It's like the classic comic strip that introduces you to the world of superheroes. Environment Modules help users and administrators manage the complex environment settings needed by different software applications. Think of it as having the ability to switch between being a detective in a noir thriller to a space explorer in a sci-fi epic, all without changing libraries. With a simple command, you can load, unload, and switch between different software packages, ensuring that your HPC adventure is both thrilling and hassle-free.

LMOD: The Modern Reboot

Next up, LMOD, a modern reboot of the classic, bringing new powers and a sleek interface to the story. LMOD is like the graphic novel remake of your favorite superhero saga, with better graphics and more powers. It's designed for the modern world of HPC, offering dynamic adjustment of the user environment via modulefiles. With LMOD, navigating through software packages becomes a smooth ride, even when dealing with intricate dependencies and versions. It's your personal guide to the multiverse of software environments, ensuring that you always have the right tools at your fingertips.

Other Mystical Tomes: Beyond the Mainstream

While Environment Modules and LMOD are the headliners, the library of HPC contains other mystical tomes, lesser-known but equally powerful. These include tools like EasyBuild and Spack, which not only manage environments but also automate the building and installation of software, adding another layer of convenience to your HPC journey.

Embarking on Your Quest

Now that you've been introduced to the magical library of module files in HPC, you're ready to embark on your quest. Remember, the power to navigate through complex software environments is now at your command. Whether you're summoning the classic spells of Environment Modules or invoking the modern magics of LMOD, your journey in the world of HPC will be filled with discoveries and achievements.

With this knowledge, you're not just stepping into the world of HPC; you're flying into it on a magic carpet, ready to explore, discover, and create. So, let's get those spells ready and start our adventure in high-performance computing!

Simple Module File for TensorFlow in Environment Modules

Diving into the mystical realm of HPC module management, let's craft some enchantments for summoning TensorFlow, a powerful ally in the quest for machine learning. We'll compare simple and complex module files within the venerable scrolls of Environment Modules and the modern grimoire of LMOD.

Environment Modules: TensorFlow/2.3.0

```
#%Module1.0
## Simple TensorFlow modulefile
proc ModulesHelp { } {
    global version
    puts stderr "TensorFlow - version $version"
}d
module-whatis "Loads TensorFlow 2.3.0 environment"

set version 2.3.0
set root /opt/apps/tensorflow/$version

prepend-path PATH $root/bin
prepend-path LD_LIBRARY_PATH $root/lib
prepend-path PYTHONPATH $root/python
```

Environment Modules Scroll: TensorFlow/2.3.0 - Explanation

This spell, written in the ancient TCL language, is straightforward. It adjusts the `PATH`, `LD_LIBRARY_PATH`, and `PYTHONPATH` to include TensorFlow's binaries, libraries, and Python bindings, respectively. Perfect for apprentices needing basic TensorFlow powers.

LMOD: TensorFlow/2.3.0

```
-- -*- lua -*-
help([[
This module loads TensorFlow 2.3.0 with its dependencies into your environment.
]])

whatis("Name: TensorFlow")
whatis("Version: 2.3.0")
whatis("Category: library, machine learning")
whatis("Keywords: Machine Learning, TensorFlow")
whatis("URL: https://www.tensorflow.org")

-- Dependencies
depends_on("python/3.8.5", "cuda/10.2", "cudnn/7.6")

-- Environment Paths
local tensorflow_root = "/opt/apps/tensorflow/2.3.0"
prepend_path("PATH", pathJoin(tensorflow_root, "bin"))
prepend_path("LD_LIBRARY_PATH", pathJoin(tensorflow_root, "lib"))
prepend_path("PYTHONPATH", pathJoin(tensorflow_root, "python"))

-- Set an environment variable, useful for custom TensorFlow configurations
setenv("TENSORFLOW_VERSION", "2.3.0")
```

LMOD: TensorFlow/2.3.0 - Explanation

This LMOD spell is more intricate, written in the Lua language. It not only adjusts the environment paths like its Environment Module counterpart but also manages dependencies with the `depends_on` function. This allows the user to load TensorFlow along with specific versions of Python, CUDA, and cuDNN, ensuring compatibility and enhancing the user's computational rituals.

Comparison and Benefits

Environment Modules Format:

- **Language:** TCL, offering simplicity and ease of use.
- **Focus:** Primarily adjusts environment paths; suitable for simpler or more static software environments.
- **Benefit:** Ideal for straightforward module files where dependency management is either minimal or handled externally.

LMOD Modules Format:

- **Language:** Lua, which provides more flexibility and advanced features.
- **Focus:** Offers dynamic dependency management with `depends_on` and environmental customization through functions like `setenv`.
- **Benefit:** Superior for complex environments where software dependencies are intricately interwoven. Its hierarchical module management ensures that only compatible modules are loaded together, preventing conflicts.

In Summary:

Environment Modules are like the trusty old map for navigating the software environment: reliable for straightforward paths. LMOD, however, is the enchanted compass that not only shows the way but also unveils paths invisible to the naked eye, automatically adjusting for obstacles and ensuring a smooth journey through the most complex software landscapes.

Environment Modules: Nvidia HPC-X 2.17.1

Crafting a complex module file for Nvidia HPC-X, a comprehensive software suite designed for high-performance computing environments, requires careful orchestration of its components—OpenMPI, CUDA, and UCX. Let’s illuminate the path by drafting these artifacts in both the Environment Modules and LMOD languages, ensuring that users can wield the power of Nvidia HPC-X with precision and ease.

Environment Modules Scroll: NvidiaHPCX/2.17.1

```
#%Module1.0
## Nvidia HPC-X 2.17.1 modulefile
proc ModulesHelp { } {
    puts stderr "Loads Nvidia HPC-X 2.17.1 with OpenMPI, CUDA support, and UCX"
}
module-whatis "Loads Nvidia HPC-X 2.17.1 environment"

set version 2.17.1
set root /opt/hpcx/$version

prepend-path PATH $root/bin
prepend-path LD_LIBRARY_PATH $root/lib

# OpenMPI setup
prepend-path PATH $root/ompi/bin
prepend-path LD_LIBRARY_PATH $root/ompi/lib
```

```
# CUDA support
prepend-path PATH /usr/local/cuda/bin
prepend-path LD_LIBRARY_PATH /usr/local/cuda/lib64

# UCX support
prepend-path PATH $root/ucx/bin
prepend-path LD_LIBRARY_PATH $root/ucx/lib

# Environment variables for MPI compilers
setenv MPI_CC "$root/ompi/bin/mpicc"
setenv MPI_CXX "$root/ompi/bin/mpicxx"
setenv MPI_FC "$root/ompi/bin/mpifort"
```

Environment Modules Scroll: NvidiaHPCX/2.17.1 - Explination

This module, crafted in the ancient TCL dialect, meticulously sets the stage for Nvidia HPC-X, arranging the environment paths and variables to seamlessly integrate OpenMPI, CUDA, and UCX.

LMOD: Nvidia HPC-X 2.17.1

```
-- -*- lua -*-
-- Written by the LMOD sorcerer
help([[
This module deploys Nvidia HPC-X 2.17.1 with OpenMPI, CUDA, and UCX support into your environment.
]])

whatis("Name: Nvidia HPC-X")
whatis("Version: 2.17.1")
whatis("Category: library, MPI, CUDA, UCX")
whatis("Keywords: HPC, Nvid
ia, MPI, CUDA, UCX")
whatis("URL: https://developer.nvidia.com/hpc-x")

-- Dependencies
depends_on("cuda/11.2")
depends_on("ucx/1.10.1")

-- Nvidia HPC-X Paths
local hpcx_root = "/opt/hpcx/2.17.1"
prepend_path("PATH", pathJoin(hpcx_root, "bin"))
prepend_path("LD_LIBRARY_PATH", pathJoin(hpcx_root, "lib"))
```

```
-- OpenMPI setup
prepend_path("PATH", pathJoin(hpcx_root, "ompi/bin"))
prepend_path("LD_LIBRARY_PATH", pathJoin(hpcx_root, "ompi/lib"))

-- CUDA setup is handled via the cuda module dependency

-- UCX setup
prepend_path("PATH", pathJoin(hpcx_root, "ucx/bin"))
prepend_path("LD_LIBRARY_PATH", pathJoin(hpcx_root, "ucx/lib"))

-- MPI compilers
setenv("MPI_CC", pathJoin(hpcx_root, "ompi/bin/mpicc"))
setenv("MPI_CXX", pathJoin(hpcx_root, "ompi/bin/mpicxx"))
setenv("MPI_FC", pathJoin(hpcx_root, "ompi/bin/mpifort"))
```

LMOD: NvidiaHPCX/2.17.1 - Explanation

In this LMOD spell, written in the Lua script, the power of Nvidia HPC-X is conjured with precision, calling upon the forces of CUDA and UCX through dependencies, and weaving them into the fabric of the user's environment with greater flexibility and specificity than its TCL counterpart.

The Arcane Comparison

Environment Modules in TCL crafts a straightforward yet effective environment, perfect for when the paths to power are clear and direct. It's akin to casting a spell with well-known incantations, suitable for many but not all mystical endeavors.

LMOD, with its Lua essence, brings a more dynamic and versatile approach. It allows for dependencies to be explicitly managed and provides a modular architecture that adapts to complex software ecosystems. It’s the equivalent of invoking ancient spirits to not only guide your path but also reshape the landscape to your needs.

Both scripts serve the noble purpose of enabling the arcane arts of HPC, yet LMOD’s ability to dynamically manage complex dependencies and environments makes it especially potent for wizards facing the multifaceted challenges of modern computational alchemy.