How Collections work & ml spider output

Robert McLay

Jan. 11, 2022

Collections

- What: A saved list of modules that can be restored
- Rules on what gets loaded.
- How collections work
- General Principle: Restoring collection is the same as by hand
- Conclusions
- Future Topics

Basics

- As many named collections as you like.
- A collection named “default” overrides site default (assuming correct startup setup)
- Collection replace current modules not an addition.
- `module load foo/1.1` always load `foo/1.1` independent of current default.
- `module load foo` always restores the default `foo`
- Unless LMOD_PIN_VERSION=yes is set.
- In this case Lmod restore module version when stored.

Story: How collections started.

- A colleague, Bill Barth, asked if there could be a way to save the current list of modules.
- The default modules were not what I wanted.
- This idea sounded simple to implement but …
- There be dragons in that idea.
- It took more than a year to work out the "right way" to do it.

Original Implementation for Collections

- Save the module table into a file (∼/.lmod.d/default)
- Restore steps:
  1. Purge ALL modules (including sticky ones)
  2. set $MODULEPATH to one stored in collection
  3. Loop over list of modules in collection (They are in load order)
  4. Remove module not in the list (DRAGONS!!)
- This does work right in certain cases.

Why this does not work

- Assume simple 4 module system: Meta, icc, impi, openmpi
- Default module: Meta (which loads icc, impi)
- User collection: Meta, icc, openmpi
- Assume no family() functions in use

Why this does not work (II)

```
User does this:
   $ ml purge; ml Meta; ml -impi openmpi; ml save
Original Collection impl:
   3a) load Meta -> load Meta, icc impi
   3b) load icc (again)
   3c) load openmpi
   4a) unload impi
```

Why this does not work (III)

- At our site, both icc and openmpi set $MPI_MODE
- $MPI_MODE is used in our local mpirun command
- Step 4a unload impi which unset $MPI_MODE
- Disaster!!! ⇒ User cannot launch mpi programs.

Why this does not work (IV)

- If any modules share an env. var. ⇒ TROUBLE!
- The problem is that setenv() is not pushenv()
- Thought about making all setenv() work like pushenv()
- Ultimately came up with a different design.
- It took several iterations to get here.

Current collection restore implementation

1. Purge ALL modules
2. Load all modules in list order BUT ignore load() like functions inside a modulefile.

How does this help?

```
3a) load Meta
   -> load Meta only (ignore load("icc", "impi"))
3b) load icc
3c) load openmpi

=> no modules to unload!!
```

What are the drawbacks to the solution?

- This works fine until “Meta” gets another “load()”
- In collections, all load() are ignored, the user doesn’t get the same modules
- Solution: error out when modules get new load() or changes via prepend_path() or append_path() to $MODULEPATH.

Saving modules in collections

- Saving causes a “show” for each module
- But only for load() like functions and changes to $MODULEPATH
- All other Lmod functions are ignored.
- A sha1 of the resulting string is computed and saved.

Restoring a collection

1. Lmod purges all modules
2. set $MODULEPATH to one stored in collection
3. Loop over list of modules in collection (They are in load order)
4. Compute sha1 of each module with only load() and changes to $MODULEPATH shown
5. Compare sha1 value with stored value in collection
6. Error out if sha1 values do not match.

Drawbacks to this solution

- Most modules are null strings
- But Meta, compiler and mpi modules changes can trigger invalid collection ⇒ rebuild collection message.

Issue 388: Forcibly loading a collection

- The drawback to users is that their job might die when a "Meta" module changes
- Issue 338 (2018) requested that a collection be loaded anyway.
- I won't do it because it won't be the same modules.

Conclusions

- Collections provide a convenient way to group modules together
- Rather than users creating their own “Meta” modules
- Collection have to be rebuilt as “Meta” modules change.

New Topic: Issue 551: Handling Descriptions: ml spider GROMACS/2019

- Level 2 output shows module help and description
- Lmod picks one modules's help and description.
- Might be confusing when there are GPU and CPU versions.
- Lmod is only going to pick one.
- Comments?

Future Topics

- Lmod Testing System?
- Explain how to pass module info to hooks(Issue 552)
- More internals of Lmod?
- Guest Presentation of special issues?