Hashdist – A tool for building and managing your scientific software distribution(s)

Chris Kees (US Army ERDC) and the HashDist Developers

Coastal and Hydraulics Laboratory
U.S. Army ERDC

Sage Days 58, June 16-20, 2014

http://github.com/hashdist

Acknowledgements

- Dag Sverre Seljebotn, Ondřej Čertík, Aron Ahmadia, Robert Maynard, Jimmy Tang, Lane Wittgen,...
- Dag Sverre's work for Simula Innovation was funded by a grant from the International Research Office, US Army Engineer Research and Development Center (BAA contract W911NF-12-1-0604)
- Ondřej's work for UT was funded by the US Army Engineer Research and Development Center (BAA Contract BAA12-4941)
- Part of Aron's work was supported by the CHL Research Participation Program administered by the Oak Ridge Institute for Science and Education

What's the problem?

- I wrote some math software and I want YOU to use it.

What's the problem?

- I wrote some math software and I want YOU to use it.
- Problem: My software needs libfoo, which you don't have.

What's the problem?

- I wrote some math software and I want YOU to use it.
- Problem: My software needs libfoo, which you don't have.
- Problem: My software needs libfoo, which you DO have in a really weird location.

What's the problem?

- I wrote some math software and I want YOU to use it.
- Problem: My software needs libfoo, which you don't have.
- Problem: My software needs libfoo, which you DO have in a really weird location.
- Problem: I built my software on an old Linux machine, you need it on OS X.

What's the problem?

- I wrote some math software and I want YOU to use it.
- Problem: My software needs libfoo, which you don't have.
- Problem: My software needs libfoo, which you DO have in a really weird location.
- Problem: I built my software on an old Linux machine, you need it on OS X.
- Problem: You need to run my software on an HPC machine.

What's the problem?

- I wrote some math software and I want YOU to use it.
- Problem: My software needs libfoo, which you don't have.
- Problem: My software needs libfoo, which you DO have in a really weird location.
- Problem: I built my software on an old Linux machine, you need it on OS X.
- Problem: You need to run my software on an HPC machine.
- Problem: You don't have root access to your machine.

What's the problem?

- I wrote some math software and I want YOU to use it.
- Problem: My software needs libfoo, which you don't have.
- Problem: My software needs libfoo, which you DO have in a really weird location.
- Problem: I built my software on an old Linux machine, you need it on OS X.
- Problem: You need to run my software on an HPC machine.
- Problem: You don't have root access to your machine.
- Solution: I need to add my software to a distribution that can handle all these cases.

Rest of the world in 2013, 2014?,...

- Debian/RedHat distributions (Linux only)

Rest of the world in 2013, 2014?,...

- Debian/RedHat distributions (Linux only)
- App store (Mac only)

Rest of the world in 2013, 2014?,...

- Debian/RedHat distributions (Linux only)
- App store (Mac only)
- Web apps (still needs good distribution)

Rest of the world in 2013, 2014?,...

- Debian/RedHat distributions (Linux only)
- App store (Mac only)
- Web apps (still needs good distribution)
- MATLAB environment (a kind of distribution)

Rest of the world in 2013, 2014?,...

- Debian/RedHat distributions (Linux only)
- App store (Mac only)
- Web apps (still needs good distribution)
- MATLAB environment (a kind of distribution)
- Modules on HPC machines.

Background

- Proteus toolkit for PDEs was building its own stack with custom scripts, makefiles, and config files
- FENICS toolkit for PDEs had its own system (Dorsal)
- Various other DoD groups had their own Python distributions
- Could possibly have added to Sage or Enthought distributions but would require many new packages and modifications to reuse “host“ packages (e.g. vendor MPI)

Requirments for new system

- Must support mixing source and “host” packages (e.g. vendor MPI)
- Must support version control and reproducibility across users
- Must support many kinds of packages (Python, C++, Fortran,...)
- Must support needs of developers (building from Git repos, trying new packages,...)

Scientific code

- Dependencies are bad: Hard to build, move between clusters

Scientific code

- Dependencies are bad: Hard to build, move between clusters
- Dependencies are good: Can build on works of others

Scientific code

- Dependencies are bad: Hard to build, move between clusters
- Dependencies are good: Can build on works of others
- Often happens: “Large frameworks” bundling things together

Scientific code

- Dependencies are bad: Hard to build, move between clusters
- Dependencies are good: Can build on works of others
- Often happens: “Large frameworks” bundling things together
  - PETSc and Trilinos for solving PDEs

Scientific code

- Dependencies are bad: Hard to build, move between clusters
- Dependencies are good: Can build on works of others
- Often happens: “Large frameworks” bundling things together
  - PETSc and Trilinos for solving PDEs
  - As soon as you want to push boundaries there’s a lot of dirty work ahead

What makes scientific codes so special?

- No root access

What makes scientific codes so special?

- No root access
- Sometimes you **need** the very latest version

What makes scientific codes so special?

- No root access
- Sometimes you **need** the very latest version
- Fortran/C++ instead of C/Java/.NET

What makes scientific codes so special?

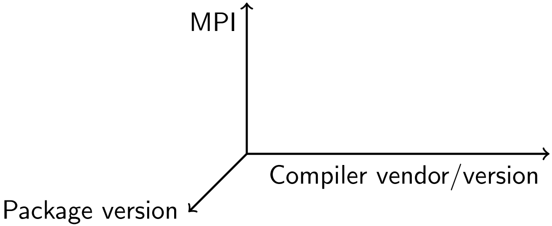
- No root access
- Sometimes you **need** the very latest version
- Fortran/C++ instead of C/Java/.NET
- Intersection of “need speed” and “do not pay dedicated application sysadmins”

Combinatorial explosion

```
/cluster/software/VERSIONS/hdf5-1.6.1/lib/libhdf5.so
/cluster/software/VERSIONS/hdf5-1.6.1_intel/lib/libhdf5.so
/cluster/software/VERSIONS/hdf5-1.6.1_pgi/lib/libhdf5.so
/cluster/software/VERSIONS/hdf5-1.8.9/lib/libhdf5.so
/cluster/software/VERSIONS/hdf5-1.8.9_intel/lib/libhdf5.so
/cluster/software/VERSIONS/hdf5-1.8.9_pgi/lib/libhdf5.so
```

Combinatorial explosion

```
/cluster/software/VERSIONS/hdf5-1.6.1/lib/libhdf5.so
/cluster/software/VERSIONS/hdf5-1.6.1_intel/lib/libhdf5.so
/cluster/software/VERSIONS/hdf5-1.6.1_pgi/lib/libhdf5.so
/cluster/software/VERSIONS/hdf5-1.8.9/lib/libhdf5.so
/cluster/software/VERSIONS/hdf5-1.8.9_intel/lib/libhdf5.so
/cluster/software/VERSIONS/hdf5-1.8.9_pgi/lib/libhdf5.so
```



MPI

Compiler vendor/version

Package version

× LAPACK × FFT library × IDL/Python version...

Established options elsewhere

- Debian/Ubuntu/RedHat/Gentoo

Established options elsewhere

- Debian/Ubuntu/RedHat/Gentoo
  - Need root

Established options elsewhere

- Debian/Ubuntu/RedHat/Gentoo
  - Need root
  - Deals poorly with combinatorial explosion

Established options elsewhere

- Debian/Ubuntu/RedHat/Gentoo
  - Need root
  - Deals poorly with combinatorial explosion
- MacPorts, Fink

Established options elsewhere

- Debian/Ubuntu/RedHat/Gentoo
  - Need root
  - Deals poorly with combinatorial explosion
- MacPorts, Fink
  - Mac only

Established options elsewhere

- Debian/Ubuntu/RedHat/Gentoo
  - Need root
  - Deals poorly with combinatorial explosion
- MacPorts, Fink
  - Mac only
  - Deals “not well enough” with combinatorial explosion

Established options elsewhere

- Debian/Ubuntu/RedHat/Gentoo
  - Need root
  - Deals poorly with combinatorial explosion
- MacPorts, Fink
  - Mac only
  - Deals “not well enough” with combinatorial explosion
- HPC environment modules

Established options elsewhere

- Debian/Ubuntu/RedHat/Gentoo
  - Need root
  - Deals poorly with combinatorial explosion
- MacPorts, Fink
  - Mac only
  - Deals “not well enough” with combinatorial explosion
- HPC environment modules
  - The sysadmins hate them

Established options elsewhere

- Debian/Ubuntu/RedHat/Gentoo
  - Need root
  - Deals poorly with combinatorial explosion
- MacPorts, Fink
  - Mac only
  - Deals “not well enough” with combinatorial explosion
- HPC environment modules
  - The sysadmins hate them
  - The users need newer/their own libraries

Current solutions

- Build yourself from source

Current solutions

- Build yourself from source
  - `./configure --prefix=$HOME/local; make; make install`

Current solutions

- Build yourself from source
  - `./configure --prefix=$HOME/local; make; make install`
  - `export LD_LIBRARY_PATH=~sigurdkn/local/lib`

Current solutions

- Build yourself from source
  - `./configure --prefix=$HOME/local; make; make install`
  - `export LD_LIBRARY_PATH=~sigurdkn/local/lib`
  - Fragile, duplication of work, some packages are very difficult to build...

Current solutions

- Build yourself from source
  - `./configure --prefix=$HOME/local; make; make install`
  - `export LD_LIBRARY_PATH=~sigurdkn/local/lib`
  - Fragile, duplication of work, some packages are very difficult to build...
- Aha! Make a script to build from source...
  - Simula: dorsal
  - 5 different within DoD (incl. python-hpcmp)
  - Sage
  - Bilder (`http://sourceforge.net/p/bilder/wiki/Home/`)
  - Lots of others ("everybody" does it)

Current solutions

- Build yourself from source
  - `./configure --prefix=$HOME/local; make; make install`
  - `export LD_LIBRARY_PATH=~sigurdkn/local/lib`
  - Fragile, duplication of work, some packages are very difficult to build...
- Aha! Make a script to build from source...
  - Simula: dorsal
  - 5 different within DoD (incl. python-hpcmp)
  - Sage
  - Bilder (`http://sourceforge.net/p/bilder/wiki/Home/`)
  - Lots of others ("everybody" does it)
- Easy to do oneself ⇒ difficult for one to get momentum

Current solutions

- Build yourself from source
  - `./configure --prefix=$HOME/local; make; make install`
  - `export LD_LIBRARY_PATH=~sigurdkn/local/lib`
  - Fragile, duplication of work, some packages are very difficult to build...
- Aha! Make a script to build from source...
  - Simula: dorsal
  - 5 different within DoD (incl. python-hpcmp)
  - Sage
  - Bilder (`http://sourceforge.net/p/bilder/wiki/Home/`)
  - Lots of others ("everybody" does it)
- Easy to do oneself ⇒ difficult for one to get momentum
- The details are different for everybody

Curation

- Debian, RedHat, cluster sysadmins, dorsal are all about curated software stacks

Curation

- Debian, RedHat, cluster sysadmins, dorsal are all about curated software stacks
- Perhaps you want 60% curated, 20% bleeding edge or manually tweaked, 20% your own code...

Curation

- Debian, RedHat, cluster sysadmins, dorsal are all about curated software stacks
- Perhaps you want 60% curated, 20% bleeding edge or manually tweaked, 20% your own code...
- The community using and supporting a distribution is as important as most of the other details.

HashDist Theory

Stateful vs. Functional

- Our software environment is usually defined by the STATE of the file system

Stateful vs. Functional

- Our software environment is usually defined by the STATE of the file system
- If I change the source or options for libfoo, then I rebuild it in /usr/lib/libfoo.so.

Stateful vs. Functional

- Our software environment is usually defined by the STATE of the file system
- If I change the source or options for libfoo, then I rebuild it in /usr/lib/libfoo.so.
- Stateful systems are fine when you control everything (a Linux distribution, Sage distribution)

Stateful vs. Functional

- Our software environment is usually defined by the STATE of the file system
- If I change the source or options for libfoo, then I rebuild it in /usr/lib/libfoo.so.
- Stateful systems are fine when you control everything (a Linux distribution, Sage distribution)
- An alternative view is to think of the build stage as a mapping from the input domain (input space $=$ dependency states $\oplus$ source code states $\oplus$ option states) to a range containing binary "artifacts" (output space of "coordinates" of unique libfoo.a's).

Stateful vs. Functional

- Our software environment is usually defined by the STATE of the file system
- If I change the source or options for libfoo, then I rebuild it in /usr/lib/libfoo.so.
- Stateful systems are fine when you control everything (a Linux distribution, Sage distribution)
- An alternative view is to think of the build stage as a mapping from the input domain (input space = dependency states $\oplus$ source code states $\oplus$ option states) to a range containing binary "artifacts" (output space of "coordinates" of unique libfoo.a's).
- Main idea: create build artifact coordinates by hashing the input state and store the build artifacts in a "database" keyed on those hashes.

Stateful vs. Functional

- Our software environment is usually defined by the STATE of the file system
- If I change the source or options for libfoo, then I rebuild it in /usr/lib/libfoo.so.
- Stateful systems are fine when you control everything (a Linux distribution, Sage distribution)
- An alternative view is to think of the build stage as a mapping from the input domain (input space = dependency states $\oplus$ source code states $\oplus$ option states) to a range containing binary "artifacts" (output space of "coordinates" of unique libfoo.a's).
- Main idea: create build artifact coordinates by hashing the input state and store the build artifacts in a "database" keyed on those hashes.
- Using the hash idea we can treat package building in a functional manner rather than a stateful manner (idea comes from Eelco Doltstra/Nix)

Cryptographic/secure hashing

- Large key size (160, 256, or 512 bits)
  - NIST standards: SHA-1, SHA-2 (224, 256, 384, 512 bits), SHA-3

Cryptographic/secure hashing

- Large key size (160, 256, or 512 bits)
  - NIST standards: SHA-1, SHA-2 (224, 256, 384, 512 bits), SHA-3
- Mathematical research topic to make h such that no information about input is revealed

Cryptographic/secure hashing

- Large key size (160, 256, or 512 bits)
  - NIST standards: SHA-1, SHA-2 (224, 256, 384, 512 bits), SHA-3
- Mathematical research topic to make h such that no information about input is revealed

$h(\texttt{'The dark fox'}) = h(\texttt{546865206461726b20666f78}\text{ hex})$
$= \texttt{b6589fc6ab0dc82cf12099d1c2d40ab994e8410c}\text{ hex}$

Cryptographic/secure hashing

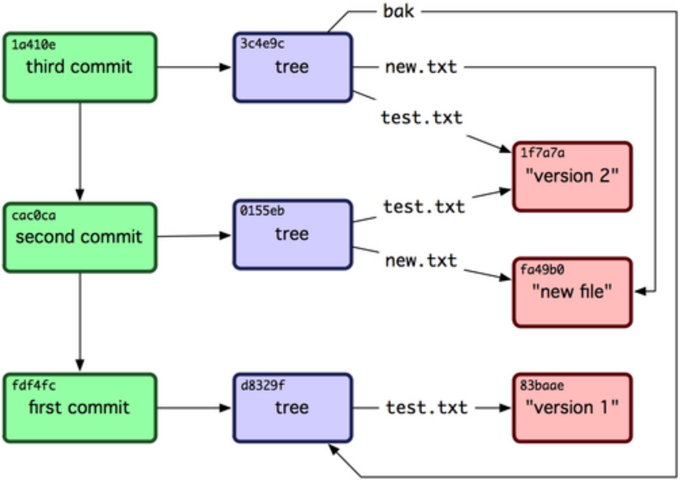
- Large key size (160, 256, or 512 bits)
    - NIST standards: SHA-1, SHA-2 (224, 256, 384, 512 bits), SHA-3
- Mathematical research topic to make h such that no information about input is revealed

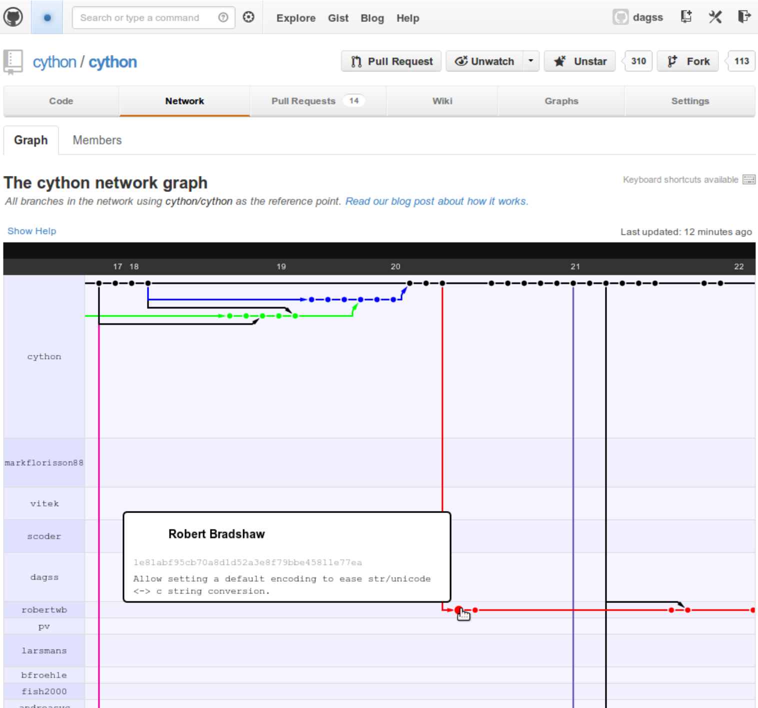
$h($`'The dark fog'`$) = h($`546865206461726b20666f67` hex$)$
$=$ `da4b9237bacccdf19c0760cab7aec4a8359010b0` hex

Example: Git

```
$ (cd code/hashdist/.git/objects; find)
./59/5a2f8e3890d0ece24514f3e32ae874f1f03ac2
./2f/780151688e1f122a5b9072d42009c80c36140c
./2f/4b2eef40b51bc2d46027d1864653b37dd05f8f
./2f/237d74e3f81f498212629ac0b96bedac4b0b36
./2f/dff799c54fed6fe96a91e1d5f1593996228ebc
./2f/27bd4efa5f8521fb98eb82181a67aae97b7f1a
./2f/3fedf882f1b28905199961356f4e00281ddf76
```

Example: Git


bak
1a410e
third commit
3c4e9c
tree
new.txt
test.txt
1f7a7a
"version 2"
cac0ca
second commit
0155eb
tree
test.txt
new.txt
fa49b0
"new file"
fdf4fc
first commit
d8329f
tree
test.txt
83baae
"version 1"



The cython network graph

All branches in the network using cython/cython *as the reference point. Read our blog post about how it works.*

Last updated: 12 minutes ago

Robert Bradshaw

1e81abf95cb70a8d1d52a3e8f79bbe45811e77ea

Allow setting a default encoding to ease str/unicode
<-> c string conversion.

Hashdist

Hash-based installation

- Linux laptop:
  `/usr/lib/libhdf5.so`

Hash-based installation

- Linux laptop:

```
/usr/lib/libhdf5.so
```

- HPC environment modules:

```
/cluster/software/VERSIONS/hdf5-1.6.1/lib/libhdf5.so
/cluster/software/VERSIONS/hdf5-1.6.1_intel/lib/libhdf5.so
/cluster/software/VERSIONS/hdf5-1.6.1_pgi/lib/libhdf5.so
/cluster/software/VERSIONS/hdf5-1.8.9/lib/libhdf5.so
/cluster/software/VERSIONS/hdf5-1.8.9_intel/lib/libhdf5.so
/cluster/software/VERSIONS/hdf5-1.8.9_pgi/lib/libhdf5.so
```

Hash-based installation

- Linux laptop:

  ```
  /usr/lib/libhdf5.so
  ```

- HPC environment modules:

  ```
  /cluster/software/VERSIONS/hdf5-1.6.1/lib/libhdf5.so
  /cluster/software/VERSIONS/hdf5-1.6.1_intel/lib/libhdf5.so
  /cluster/software/VERSIONS/hdf5-1.6.1_pgi/lib/libhdf5.so
  /cluster/software/VERSIONS/hdf5-1.8.9/lib/libhdf5.so
  /cluster/software/VERSIONS/hdf5-1.8.9_intel/lib/libhdf5.so
  /cluster/software/VERSIONS/hdf5-1.8.9_pgi/lib/libhdf5.so
  ```

  - Even `h5py-hdf5_1.8.9_pgi-python2.7-numpy1.6.3_abel_debug` incomplete

Hash-based installation

- Linux laptop:

  ```
  /usr/lib/libhdf5.so
  ```

- HPC environment modules:

  ```
  /cluster/software/VERSIONS/hdf5-1.6.1/lib/libhdf5.so
  /cluster/software/VERSIONS/hdf5-1.6.1_intel/lib/libhdf5.so
  /cluster/software/VERSIONS/hdf5-1.6.1_pgi/lib/libhdf5.so
  /cluster/software/VERSIONS/hdf5-1.8.9/lib/libhdf5.so
  /cluster/software/VERSIONS/hdf5-1.8.9_intel/lib/libhdf5.so
  /cluster/software/VERSIONS/hdf5-1.8.9_pgi/lib/libhdf5.so
  ```

  - Even `h5py-hdf5_1.8.9_pgi-python2.7-numpy1.6.3_abel_debug` incomplete
  - Abel is new and small; US Army clusters have 20-30 versions of every package.

Hash-based installation

- Linux laptop:

  `/usr/lib/libhdf5.so`
- HPC environment modules:

  ```
  /cluster/software/VERSIONS/hdf5-1.6.1/lib/libhdf5.so
  /cluster/software/VERSIONS/hdf5-1.6.1_intel/lib/libhdf5.so
  /cluster/software/VERSIONS/hdf5-1.6.1_pgi/lib/libhdf5.so
  /cluster/software/VERSIONS/hdf5-1.8.9/lib/libhdf5.so
  /cluster/software/VERSIONS/hdf5-1.8.9_intel/lib/libhdf5.so
  /cluster/software/VERSIONS/hdf5-1.8.9_pgi/lib/libhdf5.so
  ```

  - Even `h5py-hdf5_1.8.9_pgi-python2.7-numpy1.6.3_abel_debug` incomplete
  - Abel is new and small; US Army clusters have 20-30 versions of every package.
- Hashdist:

  ```
  ~/.hashdist/bld/hdf5/pe7l56vsgg43
  ~/.hashdist/bld/hdf5/tkiwsppxzc3r
  ```

  (really `hdf5/tkiwsppxzc3ro3q7pyjjxq45jgh3wwcd`)

Step 1: Hash the build

Internal protocol!

```
{
  "build" : {
    "commands" : [
      {
        "cmd" : [
          "$BASH",
          "_hashdist/build.sh"
        ]
      },
    ],
    "import" : [
      {
        "id" : "zlib/3vq2jgzdjakdhpzvpvtrbzbcrtg6etrh",
        "ref" : "ZLIB"
      }
    ]
  },
  "name" : "hdf5",
  "sources" : [
    {
      "key" : "tar.bz2:l2q3vax4o42q5zriw6k243w6ax7ov4og",
```

Step 1: Hash the build

Internal protocol!

```
{
  "build" : {
    "commands" : [
      {
        "cmd" : [
          "$BASH",
          "_hashdist/build.sh"
        ]
      },
    ],
    "import" : [
      {
        "id" : "zlib/3vq2jgzdjakdhpzvpvtrbzbcrtg6etrh",
        "ref" : "ZLIB"
      }
    ]
  },
  "name" : "hdf5",
  "sources" : [
    {
      "key" : "tar.bz2:l2q3vax4o42q5zriw6k243w6ax7ov4og",
```

Step 1: Hash the build

Internal protocol!

```
{
  "build" : {
    "commands" : [
      {
        "cmd" : [
          "$BASH",
          "_hashdist/build.sh"
        ]
      },
    ],
    "import" : [
      {
        "id" : "zlib/3vq2jgzdjakdhpzvpvtrbzbcrtg6etrh",
        "ref" : "ZLIB"
      }
    ]
  },
  "name" : "hdf5",
  "sources" : [
    {
      "key" : "tar.bz2:l2q3vax4o42q5zriw6k243w6ax7ov4og",
```

Step 2: Every build installs to separate location

Same as with the “module load“ system:

```
$ echo bld/*/* bld/adh/hwz2wmrcnbj7 bld/adh/lvpjelejrldo
bld/bzip2/zuo2tlurm7ez bld/cython/bidpqe4yx6gl bld/cython/pb
bld/cython/xoju62amg7pm bld/daetk/dpoolktbk3si
bld/docutils/ckp3lhso6vek bld/docutils/uv2667zawv7v
bld/docutils/yekwczohjna7

$ ls bld/zmq/x7ogmtwtkola/lib
pkgconfig  libzmq.a  libzmq.so  libzmq.so.3  libzmq.so.3.0.0
```

Step 2: Every build installs to separate location

Unlike “module load“ we don’t need LD_LIBRARY_PATH:

```
$ ldd bld/hdf5/tkiwsppxzc3r/lib/libhdf5.so
linux-vdso.so.1 =>  (0x00007fffe21fe000)
libpthread.so.0 => /lib/x86_64-linux-gnu/libpthread.so.0 (0x
libz.so.1 => /home/cekees/.hashdist/bld/zlib/3vq2jgzdjakd/li
libdl.so.2 => /lib/x86_64-linux-gnu/libdl.so.2 (0x00007f9d0c
libm.so.6 => /lib/x86_64-linux-gnu/libm.so.6 (0x00007f9d0cb8
libmpich.so.10 => /home/cekees/.hashdist/bld/mpich/4ztabf2u2
libc.so.6 => /lib/x86_64-linux-gnu/libc.so.6 (0x00007f9d0c35
/lib64/ld-linux-x86-64.so.2 (0x00007f9d0d9dc000)
libmpl.so.1 => /home/cekees/.hashdist/bld/mpich/4ztabf2u23u2
librt.so.1 => /lib/x86_64-linux-gnu/librt.so.1 (0x00007f9d0b
libgcc_s.so.1 => /lib/x86_64-linux-gnu/libgcc_s.so.1 (0x0000
```

Step 3: Make a profile with links

```
$ ls -la bld/profile/ycz6e4bztfgd/bin
bunzip2 -> ../../../bzip2/zuo2tlurm7ez/bin/bunzip2
bzcat -> ../../../bzip2/zuo2tlurm7ez/bin/bzcat
bzcmp -> ../../../bzip2/zuo2tlurm7ez/bin/bzcmp
...
```

Branchable software stack

-
  ```
  ~/mystack $ ls
  default.yml sources.yml build.yml abel-cluster.yml
  ```

Branchable software stack

-
  ```
  ~/mystack $ ls
  default.yml sources.yml build.yml abel-cluster.yml
  ```

-
  ```
  ~/mystack $ hit build # takes some time
  ~/mystack $ ~/local/bin/python --version
  Python 2.7.2
  ```

Branchable software stack

-
  ```
  ~/mystack $ ls
  default.yml sources.yml build.yml abel-cluster.yml
  ```

-
  ```
  ~/mystack $ hit build # takes some time
  ~/mystack $ ~/local/bin/python --version
  Python 2.7.2
  ```

-
  ```
  ~/mystack $ hit build # instant
  ```

Branchable software stack

-
  ```
  ~/mystack $ ls
  default.yml sources.yml build.yml abel-cluster.yml
  ```

-
  ```
  ~/mystack $ hit build # takes some time
  ~/mystack $ ~/local/bin/python --version
  Python 2.7.2
  ```

-
  ```
  ~/mystack $ hit build # instant
  ```

-
  ```
  ~/mystack $ git fetch from-someone-else
  ~/mystack $ git checkout someone-elses-branch
  ~/mystack $ hit build # pause for extra builds
  ~/mystack $ ~/local/bin/python --version
  Python 2.7.3
  ```

Branchable software stack

-
  ```
  ~/mystack $ ls
  default.yml sources.yml build.yml abel-cluster.yml
  ```

-
  ```
  ~/mystack $ hit build # takes some time
  ~/mystack $ ~/local/bin/python --version
  Python 2.7.2
  ```

-
  ```
  ~/mystack $ hit build # instant
  ```

-
  ```
  ~/mystack $ git fetch from-someone-else
  ~/mystack $ git checkout someone-elses-branch
  ~/mystack $ hit build # pause for extra builds
  ~/mystack $ ~/local/bin/python --version
  Python 2.7.3
  ```

-
  ```
  ~/mystack $ git checkout paper-from-2010
  ~/mystack $ hit build # instant
  ~/mystack $ ~/local/bin/python --version
  Python 2.6.3
  ```

Consequences of hash-based installation

1. More dimensions! Even `.../h5py-hdf5_1.8.9_pgi-python2.7-numpy1.6.3_debug_ubuntu12.10` is not exhaustive; "`h5py/5ffg...`" caters for everything

Consequences of hash-based installation

1. More dimensions! Even

   `.../h5py-hdf5_1.8.9_pgi-python2.7-numpy1.6.3_debug_ubuntu12.10`

   is not exhaustive; "`h5py/5ffg...`" caters for everything

2. For free: Atomic upgrades

   ```
   $ hit build
   # ...then power shuts down, but you're good
   ```

Consequences of hash-based installation

1. More dimensions! Even

   `.../h5py-hdf5_1.8.9_pgi-python2.7-numpy1.6.3_debug_ubuntu12.10`

   is not exhaustive; "`h5py/5ffg...`" caters for everything
2. For free: Atomic upgrades

   ```
   $ hit build
   # ...then power shuts down, but you're good
   ```
3. Much easier: Uninstall (mark&sweep rather than file tracking)

Consequences of hash-based installation

1. More dimensions! Even
   `.../h5py-hdf5_1.8.9_pgi-python2.7-numpy1.6.3_debug_ubuntu12.10`
   is not exhaustive; "`h5py/5ffg...`" caters for everything
2. For free: Atomic upgrades

   ```
   $ hit build
   # ...then power shuts down, but you're good
   ```

3. Much easier: Uninstall (mark&sweep rather than file tracking)
4. Jump around in software history (or between branches) in seconds!

Consequences of hash-based installation

1. More dimensions! Even
   `.../h5py-hdf5_1.8.9_pgi-python2.7-numpy1.6.3_debug_ubuntu12.10`
   is not exhaustive; "`h5py/5ffg...`" caters for everything
2. For free: Atomic upgrades

   ```
   $ hit build
   # ...then power shuts down, but you're good
   ```
3. Much easier: Uninstall (mark&sweep rather than file tracking)
4. Jump around in software history (or between branches) in seconds!

Sophisticated features with simple implementation

Prior art: Eelco Dolstra's PhD thesis/the Nix project

Demo

- Show a stack/distribution profile spec
- Show package spec and modify source hash (to build new artifact)
- Show package spec and modify build options (to build new artifact)
- Revert package spec to show no rebuild occurs
- Show 'hit develop' to build a “virtual env”

Status

- Hashdist library and command line interface up and running

Status

- Hashdist library and command line interface up and running
- Multiple stacks (Proteus, SciPy Stack, Clawpack) across multiple platforms (HPC, Linux, OS X, Cygwin)

Status

- Hashdist library and command line interface up and running
- Multiple stacks (Proteus, SciPy Stack, Clawpack) across multiple platforms (HPC, Linux, OS X, Cygwin)
- Soon: relocatable builds, IPython Notebook integration, flexible constraints