ROOT packaging with Conda

Chris Burr

ROOT User's Workshop 2022 ○ 10th May 2022

Image: CERN-EX-66954B © 1998-2018 CERN

What is point of this talk?

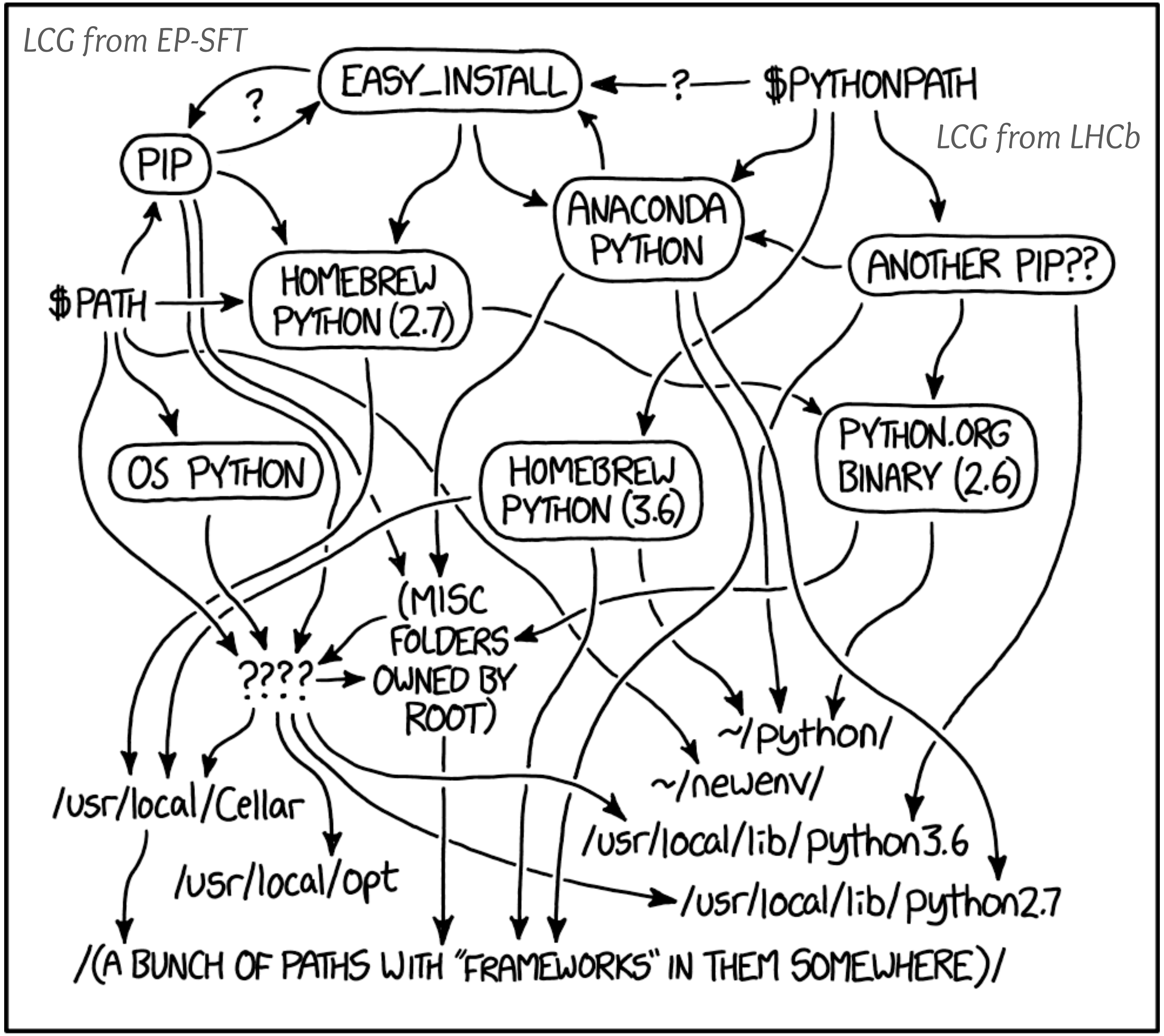
- What is package management and why should you care?

- The takeaway message depends on why you’re here:
  - If you’re a **Physicist**: Makes it easier to experiment, collaborate and preserve research
  - If you’re **Providing Support**: Enables you to guide users be more efficient
  - If you’re a **ROOT Developer**: Make ROOT more user friendly

- Why is Conda in particular useful for problems we face?

What is package management?

Well? What is the problem?


LCG from EP-SFT

LCG from LHCb

MY PYTHON ENVIRONMENT HAS BECOME SO DEGRADED THAT MY LAPTOP HAS BEEN DECLARED A SUPERFUND SITE.

https://xkcd.com/1987/

XRootD

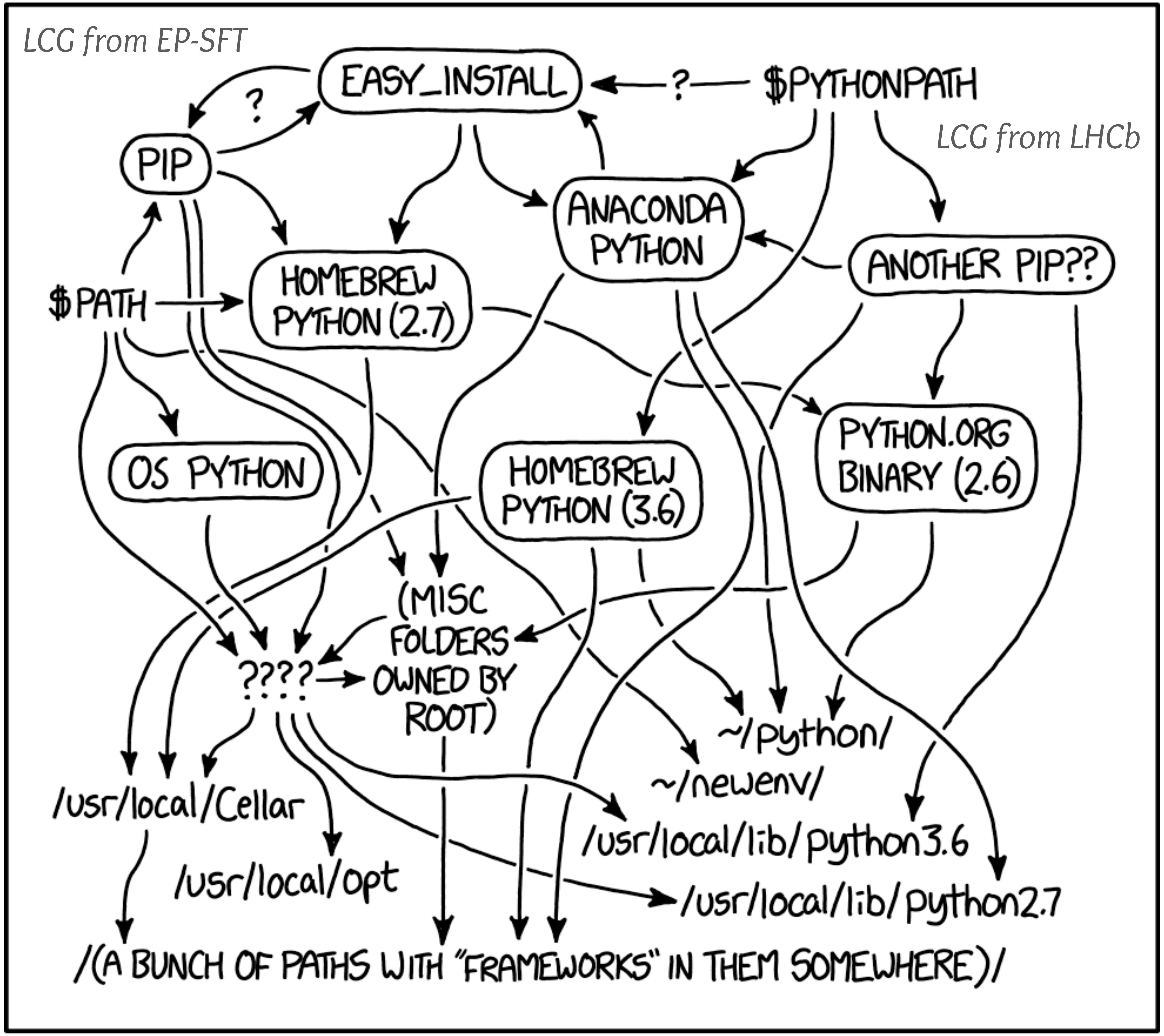
Homebrew

ROOT builtin

ROOT
Data Analysis Framework

Homebrew + Python 3.10

Self compiled against Python 3.9
(but broken since updating to 3.10 homebrew)



LCG from EP-SFT

LCG from LHCb

MY PYTHON ENVIRONMENT HAS BECOME SO DEGRADED
THAT MY LAPTOP HAS BEEN DECLARED A SUPERFUND SITE.

GCC

GCC 7.3
GCC9.2
GCC11

Clang 9
Clang 12
Clang 13
Clang 14

https://xkcd.com/1987/

Xeus Cling

XRootD

Homebrew
ROOT builtin

PyTorch

ROOT
Data Analysis Framework

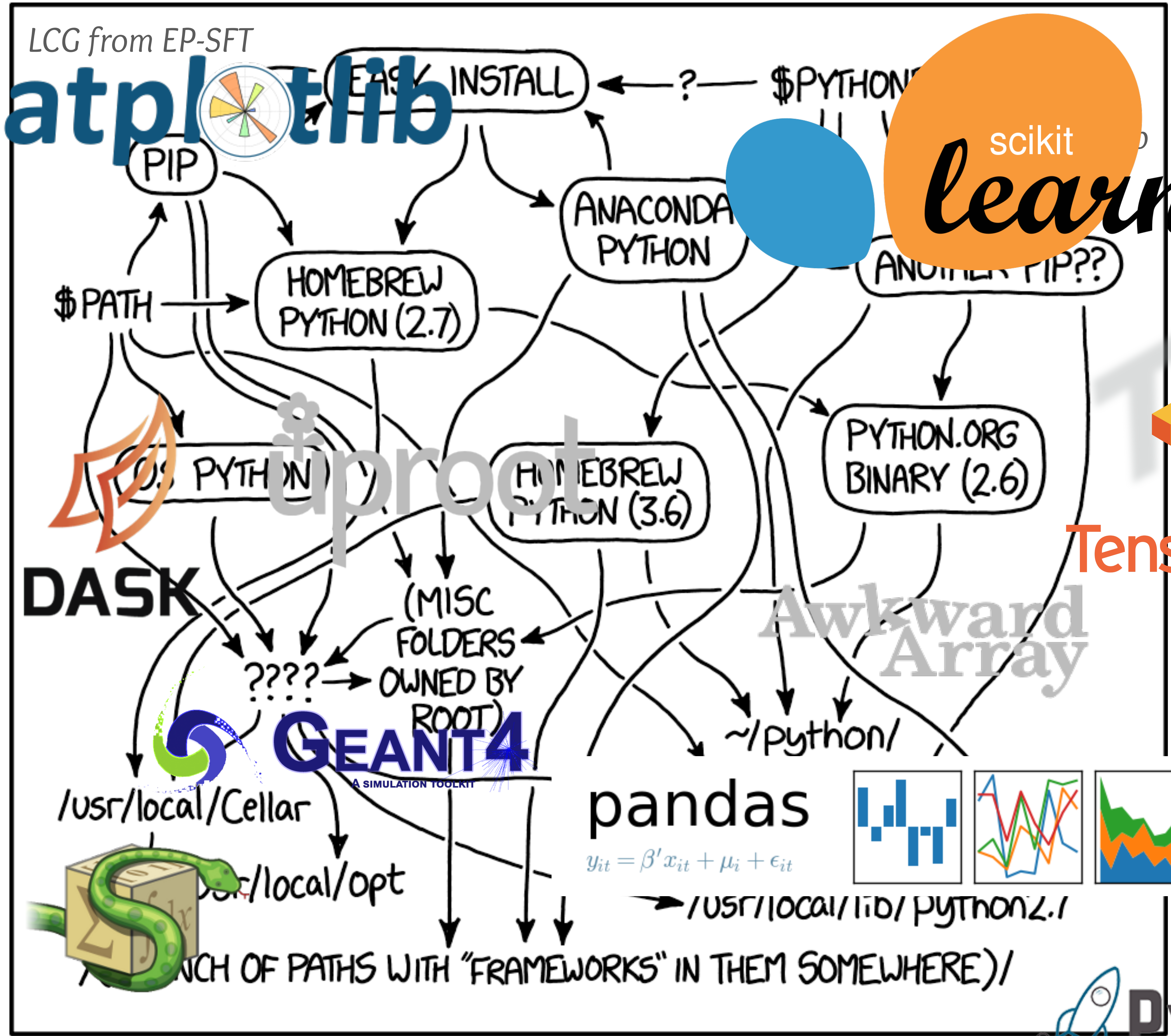
Homebrew + Python 3.10

Self compiled against Python 3.9
(but broken since updating to 3.10 homebrew)

dmlc XGBoost

LCG from EP-SFT

matplotlib



MY PYTHON ENVIRONMENT HAS BECOME SO DEGRADED THAT MY LAPTOP HAS BEEN DECLARED A SUPERFUND SITE.

scikit learn

uproot

DASK

Geant4
A SIMULATION TOOLKIT

pandas
$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$

Awkward Array

GCC

GCC 7.3
GCC9.2
GCC11

TensorFlow

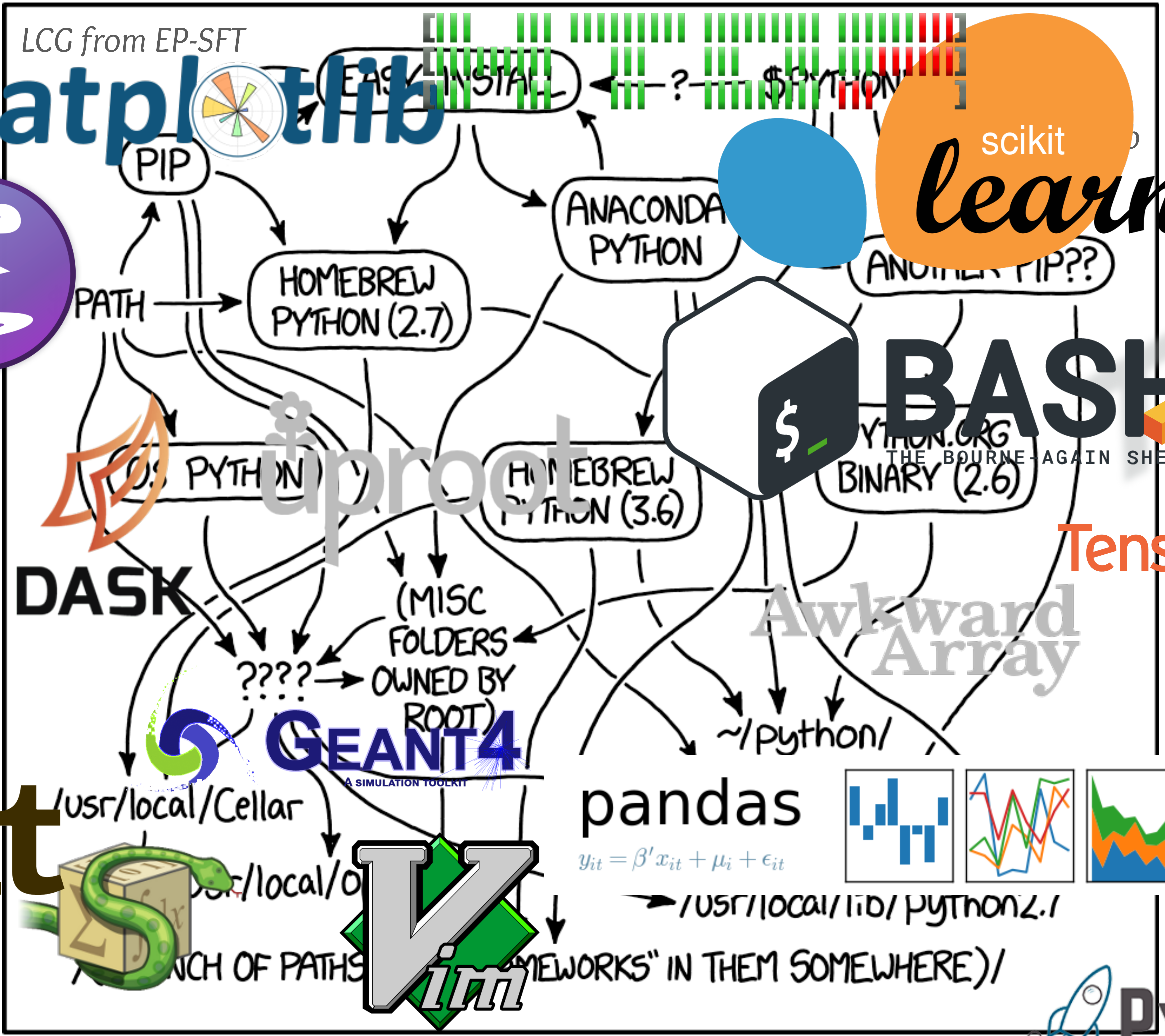
Clang 9
Clang 12
Clang 13
Clang 14

PyMC3

https://xkcd.com/1987/

Homebrew
ROOT builtin

LCG from EP-SFT



MY PYTHON ENVIRONMENT HAS BECOME SO DEGRADED THAT MY LAPTOP HAS BEEN DECLARED A SUPERFUND SITE.

Homebrew → Python 3.10

Self compiled against Python 3.
(but broken since updating to 3.10 homebrew)

GCC 7.3
GCC9.2
GCC11

Clang [illegible]
Clang 12
Clang 13
Clang 14

https://xkcd.com/1987/

Xeus Cling
XRootD
matplotlib
scikit learn
GCC
PyTorch
BASH THE BOURNE-AGAIN SHELL
TensorFlow
ROOT Data Analysis Framework
DASK
uproot
GEANT4
Awkward Array
pandas
Homebrew
git
dmlc XGBoost
PyMC3
PIP
PATH
EASY_INSTALL
HOMEBREW PYTHON (2.7)
ANACONDA PYTHON
ANOTHER PIP??
OS PYTHON
HOMEBREW PYTHON (3.6)
PYTHON.ORG BINARY (2.6)
(MISC FOLDERS OWNED BY ROOT)
????
/usr/local/Cellar
/usr/local/opt
~/python/
/usr/local/lib/python2.7
(A BUNCH OF PATHS WITH "FRAMEWORKS" IN THEM SOMEWHERE)/
MY PYTHON ENVIRONMENT HAS BECOME SO DEGRADED THAT MY LAPTOP HAS BEEN DECLARED A SUPERFUND SITE.

So...how can this be better?

What is a package manager?

- ➤ Automates the process of installing, upgrading, and removing software

- ➤ Generic package managers
  - ➤ Can provide almost any piece of software
  - ➤ **Examples:** apt-get, yum, zypper, pacman, brew, port, emerge, nix, conda…

- ➤ Application-level package managers
  - ➤ Targeted at a specific programming language or application
  - ➤ Can have specialised integrations (especially for development)
  - ➤ **Examples:** pip, cargo, yarn, npm, go, cpan, cran, ctan, maven, …

What is Conda

- ➤ ***Language agnostic*** package manager (Python, C++, R, Julia, Rust, Go, Java, Ruby, Fortran, …)
- ➤ Multi platform (Linux, macOS, Windows)
- ➤ Multi architecture (i386, x86_84, aarch64, ppc64le, partially s390x and could imagine adding others)
- ➤ Provides “environments” which are self contained sysroots in a folder
  - ➤ No admin privileges required
  - ➤ Easy to preserve long term
- ➤ Easily switch between Python versions, compilers and other packages

conda-forge

- Community maintained collection of conda packages
- Over 17,900 packages available and rapidly growing
- Over 3,900 maintainers
- Over 350,000,000 package downloads each month
- Fiscally sponsored project of NumFOCUS
  - Work with many organisations to better support use cases (e.g. nvidia with CUDA)
- Includes everything “user facing” (vim/curl/findutils/htop/…)

Allows you to combine software easily

```
mamba create --name my-analysis \
    python=3.10 ipython pandas matplotlib \
    root boost  \
    tensorflow xgboost \
    bash htop vim git
```

An environment is just a folder

Can trivially create one for each task

https://xkcd.com/1987/

An aside: conda vs mamba vs micromamba vs anaconda

- Conda-forge has grown very large, conda sometimes struggles to manage

- Mamba is an alternative implementation with a faster dependency solver
  - Will replace the default Conda solver eventually
  - Latest version of Conda supports using mamba by passing "`--experimental-solver=libmamba`"
  - For details see: A faster Conda for a growing community

- Micromamba is a small standalone binary that is mostly-compatible
  - Very helpful as a small/fast alternative for CI and containers

- Anaconda Inc. provides the commercial Anaconda and Miniconda installers
  - Not compatible with DIRAC, should make a new environment with Conda-forge
  - Or use one of the free Miniforge installers

TLDR: If Conda is slow, replace `conda install/create` **with** `mamba install/create`

- ➤ Reliably install ROOT in under 5 minutes on any* machine
  - ➤ Linux, macOS, and Windows Subsystem for Linux (ROOT doesn’t support native 64-bit Windows)
- ➤ Complete installation with C++17, graphics, OpenGL, root7
- ➤ Seamlessly integrates with the rest of conda-forge
  - ➤ No PYTHONPATH/LD_LIBRARY_PATH mess
  - ➤ Easily switch between versions of Python, ROOT and anything else
- ➤ Downloaded over 375,000 times
- ➤ Also provide a nightly build of master

```
conda create --name root-nightly-env \
    -c https://root.cern/download/conda-nightly/latest \
    root-nightly
```

*Almost

Other architectures

- Conda(-forge) has very robust support for many platforms
  - linux-64, osx-64, win-64, osx-arm64, linux-aarch64, linux-ppc64le (more are likely to appear)
  - Cona-forge has very robust infrastructure for (cross-)compiling to many architectures
- The linux-aarch64 and linux-ppc64le packages are built by cross-compiling*
- Many osx-arm64 packages were available before the release of ARM Macs
- Enabling cross-compilation is surprisingly easy (and often trivial)

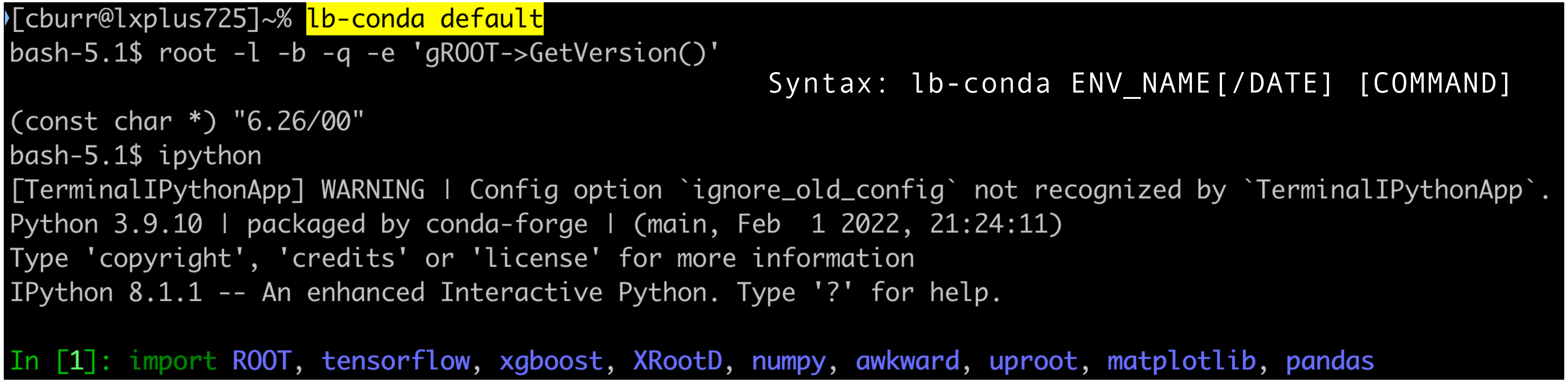
**For simplicity automatic emulation with binfmt_misc +QEMU is used to run rootcling_stage1, generate hsimple.root and run tests*

Downsides

- ➤ The main downside to Conda environments is they are often large
  - ➤ With many small files
- ➤ Works fine with SSD storage, manageable with HDD
  - ➤ Slow with AFS and unusable with EOSFUSE
  - ➤ Impractical to do on every node for large batch/grid submissions
- ➤ Ideally need a filesystem for distributing many small read-only files…

User analysis environments on CVMFS in LHCb

- Having the ability to manage arbitrary environments is nice
  - But 90% of use cases just need a recent version of popular packages
- LHCb now distributes several Conda environments on CVMFS
  - Environment activation is hidden behind a command modelled after the LHCb physics application launcher
  - Versioned by date+time and never removed for long term preservation
  - Solves the storage IOPS issues with environment creation
- Makes collaboration simpler: no need to explain how to make environments



```
[cburr@lxplus725]~% lb-conda default
bash-5.1$ root -l -b -q -e 'gROOT->GetVersion()'
                                                   Syntax: lb-conda ENV_NAME[/DATE] [COMMAND]
(const char *) "6.26/00"
bash-5.1$ ipython
[TerminalIPythonApp] WARNING | Config option `ignore_old_config` not recognized by `TerminalIPythonApp`.
Python 3.9.10 | packaged by conda-forge | (main, Feb  1 2022, 21:24:11)
Type 'copyright', 'credits' or 'license' for more information
IPython 8.1.1 -- An enhanced Interactive Python. Type '?' for help.

In [1]: import ROOT, tensorflow, xgboost, XRootD, numpy, awkward, uproot, matplotlib, pandas
```

Other uses of conda-forge in LHCb

- ➤ Providing the default LHCb computing environment (lxplus, institutes, locally)
- ➤ Distributing calibration tools
- ➤ Running the grid middleware clients and services (DIRAC)
- ➤ Running pilot jobs on the grid
- ➤ WIP: New nightly build infrastructure (replacing Jenkins)

What could be improved for distributing ROOT?

- The matrix of builds is slowly getting out of hand
  - 5 platforms x 4 Python versions = 20 builds, each taking 2+ hours
  - Enabling CUDA would ~triple this!
  - Standalone builds for PyROOT and CUDA on top of a base installation would make this much simpler

- An environment containing ROOT and it's dependencies is 2.2GB!
  - ROOT itself is 640MB
  - Having a widely agreed convention for how to split up ROOT into components would save a lot of resources

- True cross-compilation support
  - Currently no public CI providers are supporting macOS on ARM and emulation isn't possible

- Supporting multiple versions?
  - Downstream packages have to be built for each ROOT release (no ABI compatibility)
  - Could consider supporting multiple (e.g. latest patch of 6.24.x and 6.26.x)

Summary and Conclusions

- Hopefully you can see how package management is useful!

- Conda allows end users to easily manage their own software environments
  - But for larger communities central management can be a helpful addition

- Conda-forge is an friendly and open community
  - If something is missing or broken, or if you maintain a package, please consider helping out!
  - Bots take care of most of the tedious work most packages are very little effort to maintain

Thanks to everyone in the ROOT team who has helped support the maintenance of the Conda package!

How to install ROOT using conda?

1a) Download Linux and Windows Subsystem for Linux* (replace x86_64 with aarch64/ppc64le)

```
wget -nv https://github.com/conda-forge/miniforge/releases/latest/download/Mambaforge-Linux-x86_64.sh -O mambaforge.sh
```

1b) Download macOS (replace arm64 with x86_64 for Intel Macs)

```
wget -nv https://github.com/conda-forge/miniforge/releases/latest/download/Mambaforge-MacOSX-arm64.sh -O mambaforge.sh
```

2) Install conda

```
bash mambaforge.sh -b -p $HOME/mambaforge
source $HOME/mambaforge/etc/profile.d/conda.sh
conda config --set auto_activate_base false
```

← **Can be safely added to your bashrc, only provides the "conda activate" shell function**

3) Create an environment and activate it

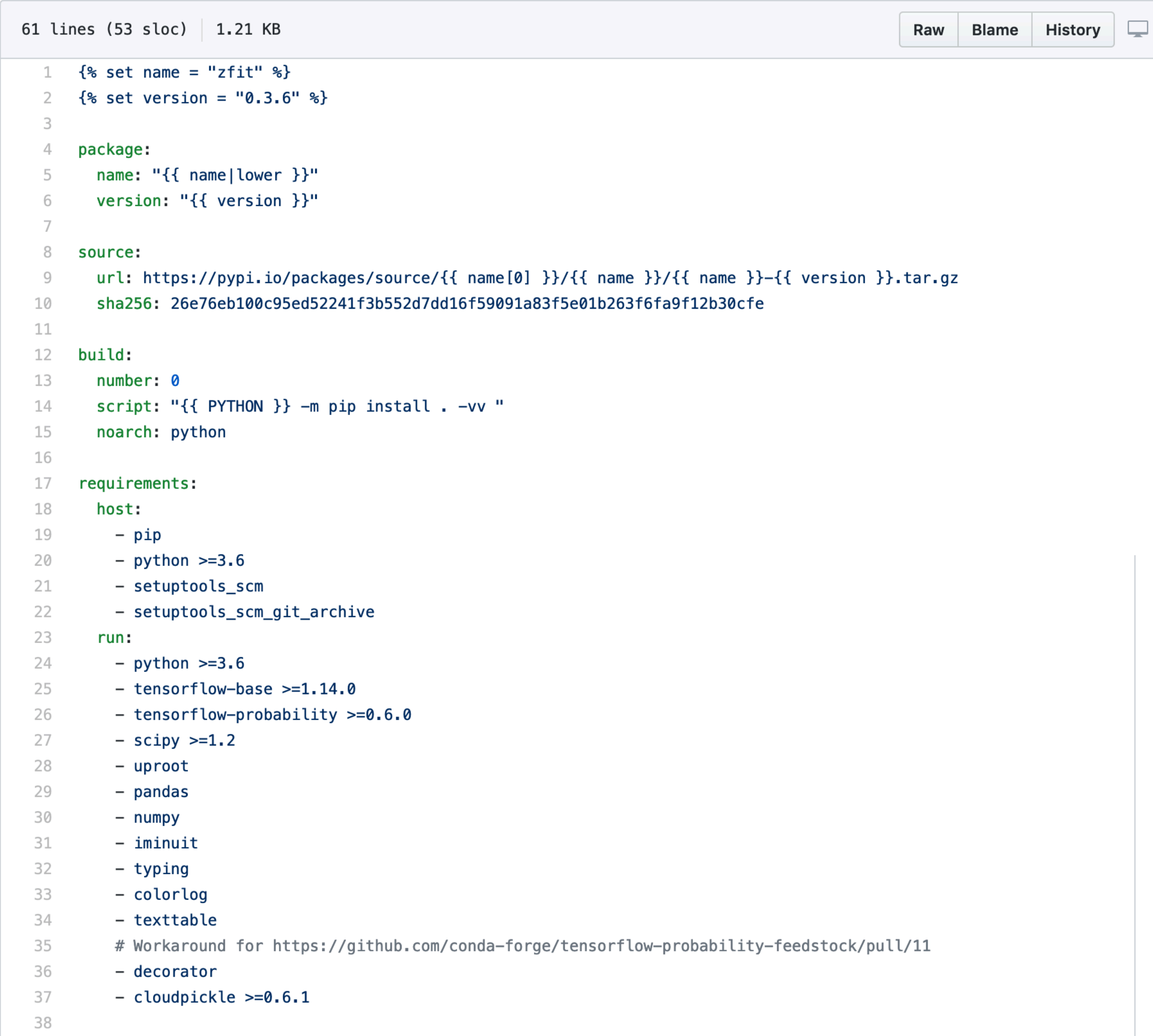
```
mamba create --name my-environment python=3.10 ipython root jupyterlab
conda activate my-environment
```

*WSL only as ROOT doesn't support native 64-bit Windows

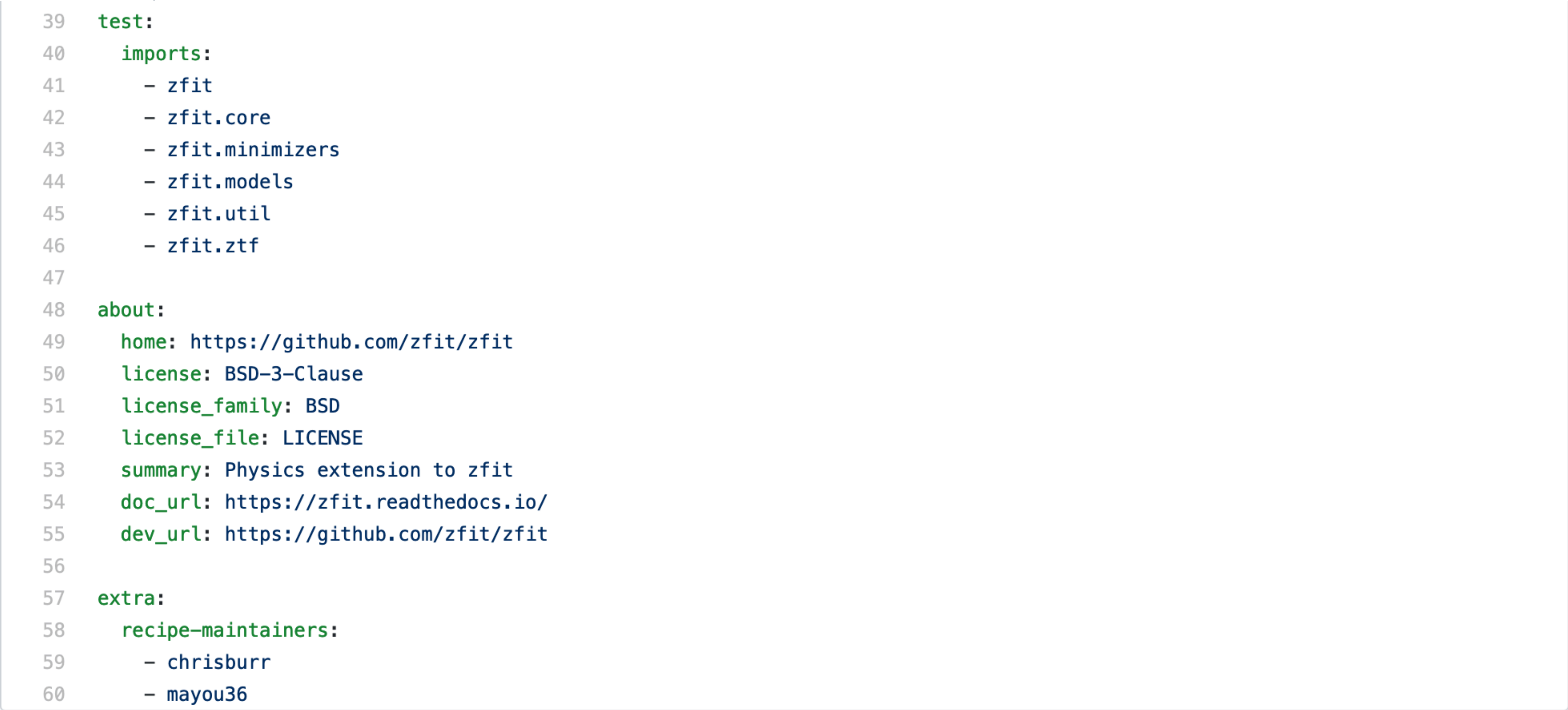
Questions?

How does conda-forge work?

Adding a new package to conda-forge



```yaml
{% set name = "zfit" %}
{% set version = "0.3.6" %}

package:
  name: "{{ name|lower }}"
  version: "{{ version }}"

source:
  url: https://pypi.io/packages/source/{{ name[0] }}/{{ name }}/{{ name }}-{{ version }}.tar.gz
  sha256: 26e76eb100c95ed52241f3b552d7dd16f59091a83f5e01b263f6fa9f12b30cfe

build:
  number: 0
  script: "{{ PYTHON }} -m pip install . -vv "
  noarch: python

requirements:
  host:
    - pip
    - python >=3.6
    - setuptools_scm
    - setuptools_scm_git_archive
  run:
    - python >=3.6
    - tensorflow-base >=1.14.0
    - tensorflow-probability >=0.6.0
    - scipy >=1.2
    - uproot
    - pandas
    - numpy
    - iminuit
    - typing
    - colorlog
    - texttable
    # Workaround for https://github.com/conda-forge/tensorflow-probability-feedstock/pull/11
    - decorator
    - cloudpickle >=0.6.1

```



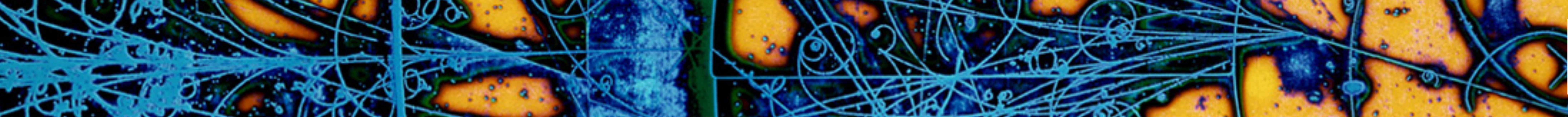
```yaml
test:
  imports:
    - zfit
    - zfit.core
    - zfit.minimizers
    - zfit.models
    - zfit.util
    - zfit.ztf

about:
  home: https://github.com/zfit/zfit
  license: BSD-3-Clause
  license_family: BSD
  license_file: LICENSE
  summary: Physics extension to zfit
  doc_url: https://zfit.readthedocs.io/
  dev_url: https://github.com/zfit/zfit

extra:
  recipe-maintainers:
    - chrisburr
    - mayou36
```

- ➤ Create a pull request against

  https://github.com/conda-forge/staged-recipes

- ➤ Can be mostly automated using

  `conda skeleton pypi zfit`



Updates

- ➤ Bots monitor for new releases
  - ➤ Even works with non-standard URLs
- ➤ Maintainers normally just have to click merge

conda-forge / phasespace-feedstock
Sponsor
Unwatch 6
Star 0
Fork 1
Code
Issues 0
Pull requests 0
Actions
Projects 0
Security
Insights
Edit
Conversation 1
Commits 2
Checks 2
Files changed 2
+3 −2
4
Viewed

phasespace v1.0.4 #1

Merged chrisburr merged 2 commits into `conda-forge:master` from `regro-cf-autotick-bot:1.0.4` 2 days ago

regro-cf-autotic... commented 2 days ago

It is very likely that the current package version for this
Notes for merging this PR:

1. Feel free to push to the bot's branch to update this
2. The bot will almost always only open one PR per ve
   Checklist before merging this PR:

- ☐ Dependencies have been updated if changed
- ☐ Tests have passed
- ☐ Updated license if changed and `license_file` is p

Note that the bot will stop issuing PRs if more than 3 Ve
open. If you don't want to package a particular version

If this PR was opened in error or needs to be updated p
The bot will close this PR and schedule another one.

recipe/meta.yaml

```
@@ -1,13 +1,13 @@
  {% set name = "phasespace" %}
- {% set version = "1.0.3" %}
+ {% set version = "1.0.4" %}

  package:
    name: "{{ name|lower }}"
    version: "{{ version }}"

  source:
    url: https://github.com/zfit/phasespace/archive/{{ version }}.tar.gz
-   sha256: ad1b322c1c47378ec6687c1ec30d5b92101c883e96bbebea345526b21596516e
+   sha256: 18e709a27111f96276aaa1f0df073e4cefc5e764ed9551de24b345aa3cc88790

  build:
    number: 0
```

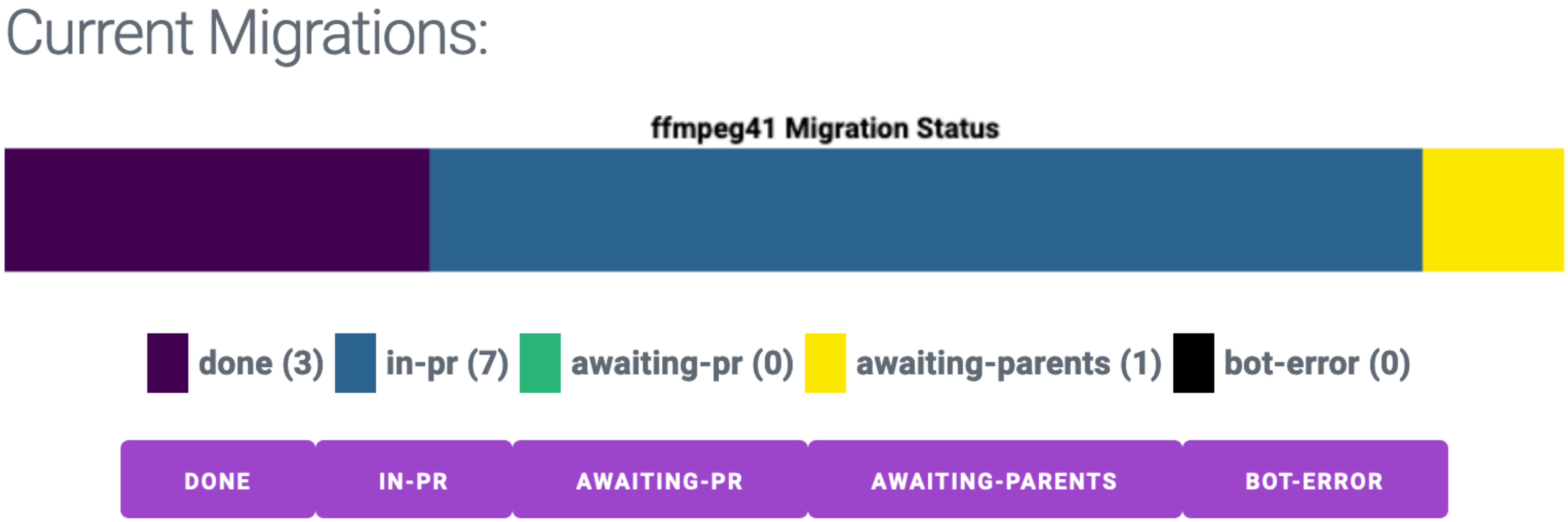
Binary only distribution

- Conda only supports installing binaries*
- Relies on the solver knowing about API/ABI compatibility
- Packages with shared libraries should specify what their ABI stability is
- Doesn’t necessarily restrict what you can do
  - Variants can be used to provide a matrix of different builds
  - BLAS can be provided by netlib, mkl, blis and openblas
  - Several MPI variants
  - TensorFlow has CPU and (several) GPU variants

**Some organisations mirror the conda-forge build infrastructure for their own internal use*

Migrations

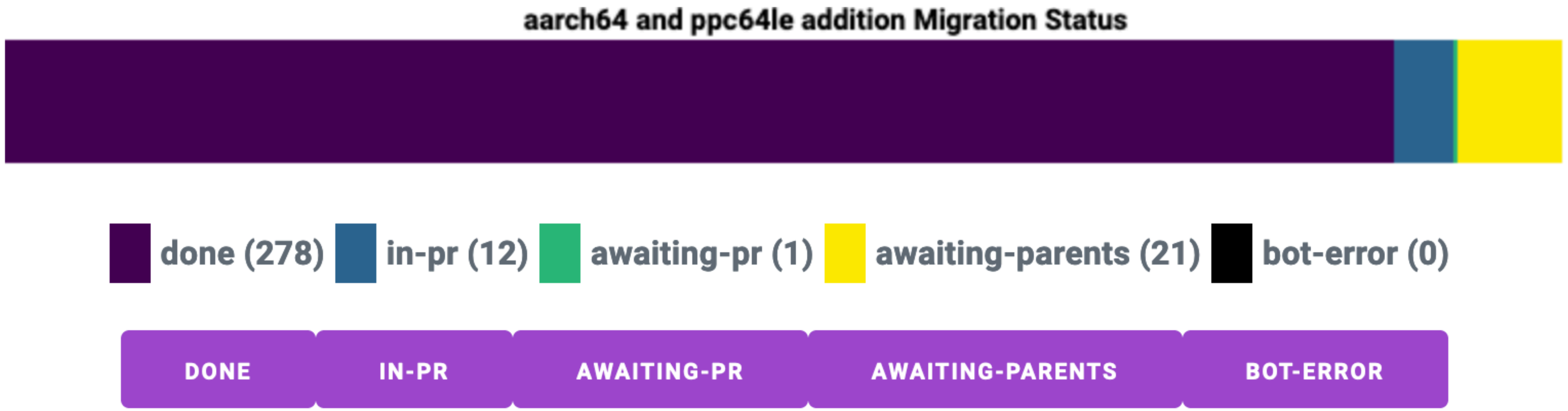
- ➤ What about when ABIs change? More 🤖!
- ➤ A line is added to a git repository
- ➤ Pull requests appear that rebuild packages in the correct order



Current Migrations:

Alternative architectures

- ➤ Migration is currently ongoing for ppc64le and aarch64 support
  - ➤ ROOT is included as a target



- ➤ Support for compiling CUDA with nvcc is rapidly maturing
  - ➤ Adds three additional additional targets (different driver versions)
  - ➤ Close to being fully supported by the conda-forge tooling
  - ➤ GPU variants of packages are already being added

How not to use pip

- Installing should be as simple as:

```
pip install PACKAGE_NAME
```

- Don’t use things that modify global state:

```
sudo pip install PACKAGE_NAME
```

  - Interacts poorly with system package managers
  - Can make it impossible to update or install packages using apt/yum/pacman/…

```
pip install --user PACKAGE_NAME
```

  - Normally has a higher priority in the Python search order
  - Can break other installations (e.g. use on lxplus can break your experiments software stack)

- **venv allow you to create environments from arbitrary Python installs**

Build infrastructure for conda-forge

- One repository per package ("feedstock")
- All packages are built using well known CI providers
- Currently mostly Azure Pipelines with Travis CI providing linux-ppc64le and linux-aarch64
- All managed by an external package: conda-smithy
  - Used to regenerate CI configuration for each update