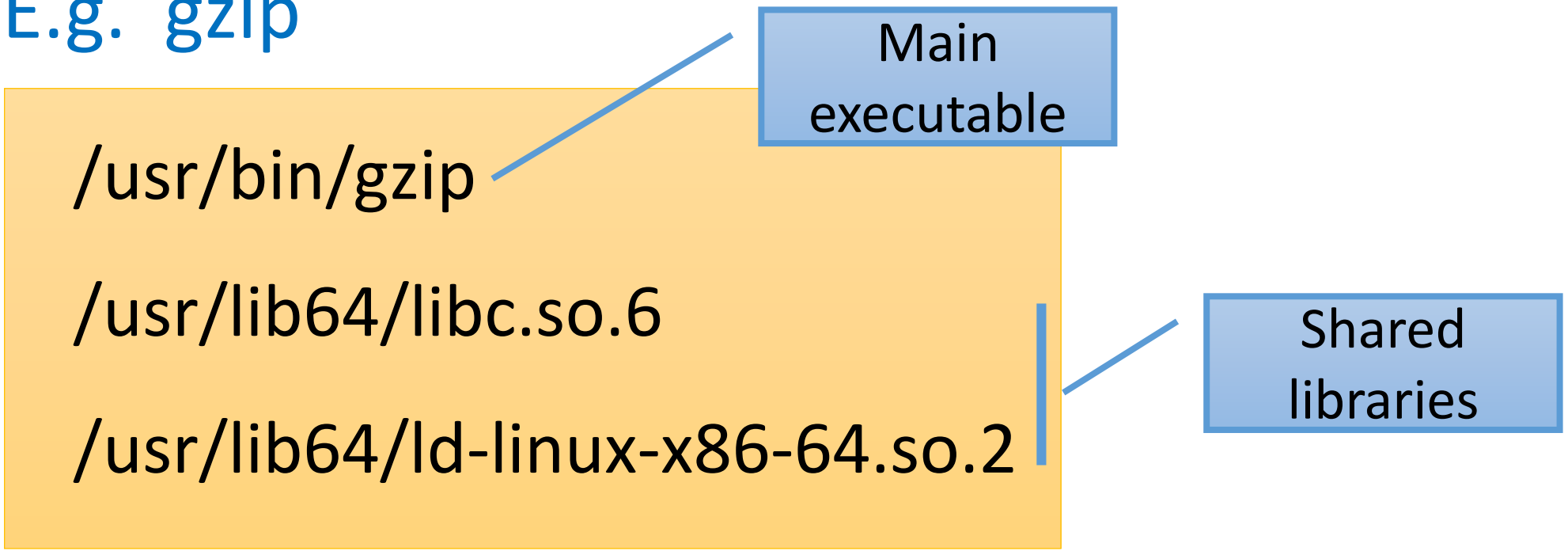
Linux Software Installation

Part 1

Qi Sun
Bioinformatics Facility

Components of software and where they are located

E.g. gzip



Software distributed with system

/usr
- bin
- lib
- lib64

Software installed on BioHPC

/programs
- bwa-0.5.9
- bwa-0.7.8
- …
- hdf5-1.8.10
- hdf5-1.10.1
- …

Extra software Installed by administrator

/usr/local
- bin
- lib
- lib64

Software installed by you

/home/xxxxx

How to install software by yourself?

/usr

/usr/local

root user can write

/programs

Only “programs” user can write

/home/xxxxx

/workdir

Directories that you have write privilege.

How to install software by yourself?

/usr

/usr/local

Default directories for installation tools

/programs

/home/xxxxx

/workdir

There are ways to tell the installation tool to put software and libraries in other directories.

How to run software installed by you

Specify path of the executable:

```
export PATH=/home/xxxxx/programs:$PATH
```

How to run software installed by you

Specify the path of libraries

C export LD_LIBRARY_PATH=/home/xxxxx/lib

PERL export PERL5LIB=/home/xxxxx/perl5/5.22.0

PYTHON export PYTHONPATH=/home/xxxxx/python/lib/python2.7/site_packages

* Once you set library path this way, they have precedence over other installed libraries with the same names.

Let’s examine a software: “Entropy”

Main executable

`/programs/entropy/bin/entropy`

Identify which library files are linked

C

```
ldd /programs/entropy/bin/entropy
```

```
linux-vdso.so.1 =>  (0x00007ffefb1d5000)
libgsl.so.0 => /lib64/libgsl.so.0 (0x00007efe3544a000)
libgslcblas.so.0 => /lib64/libgslcblas.so.0 (0x00007efe3520c000)
libz.so.1 => /lib64/libz.so.1 (0x00007efe34ff6000)
libdl.so.2 => /lib64/libdl.so.2 (0x00007efe34df2000)
libm.so.6 => /lib64/libm.so.6 (0x00007efe34aef000)
libstdc++.so.6 => /lib64/libstdc++.so.6 (0x00007efe347e7000)
libgcc_s.so.1 => /lib64/libgcc_s.so.1 (0x00007efe345d1000)
libc.so.6 => /lib64/libc.so.6 (0x00007efe3420d000)
libpthread.so.0 => /lib64/libpthread.so.0 (0x00007efe33ff1000)
libsatlas.so.3 => /usr/lib64/atlas/libsatlas.so.3 (0x00007efe333a4000)
/lib64/ld-linux-x86-64.so.2 (0x0000556c6875e000)
libgfortran.so.3 => /lib64/libgfortran.so.3 (0x00007efe33081000)
libquadmath.so.0 => /lib64/libquadmath.so.0 (0x00007efe32e45000)
```

To run the software, you need to specify the path of the main code and extra libraries not in standard library paths:

```
export PATH=/programs/entropy/bin:$PATH

export LD_LIBRARY_PATH=/programs/gcc-
5.3.0/lib:/programs/gcc-5.3.0/lib64
```

- You need to do this every time you start a new “ssh” session;
- Alternatively, put the lines in /home/xxxxx/.bashrc file, which is run automatically when a session starts.

A tool to examine which executable file is used:

which bwa

/programs/bin/bwa/bwa

echo $PATH
/programs/docker/bin:/usr/local/bin:/usr/bin:/usr/local/sbin:/usr/sbin:/programs/bin/mummer:/programs/bin/util:/programs/bin/bowtie:/programs/bin/bwa:/programs/bin/cufflinks:/programs/bin/samtools:/programs/bin/tophat:/programs/bin/fastx:/programs/bin/blast:/programs/bin/igv:/programs/bin/velvet:/programs/bin/iAssembler:/programs/bin/GATK:/programs/bin/454:/programs/bin/blat:/programs/bin/perlscripts……

Types of Software: Script vs Binary

Binary: C

Script: PERL, R, BASH

Bytecode: JAVA,PYTHON

A script is a text file, and it requires an interpreter software to run.

Run script (In the example, it is a python script, and requires python software to interpret)

```
python /programs/pybedtools/bin/intron_exon_reads.py
```

Run binary software (does not need another interpreter software)

```
/programs/ bwa-0.7.13/bwa
```

Scripts can be run without interpreter name in the command line:

/programs/pybedtools/bin/**intron_exon_reads.py**

Instead of

python /programs/pybedtools/bin/**intron_exon_reads.py**

Two requirements to do this:

a. Shebang line is present in the script;
b. The file needs to be executable.

The **Shebang** line is the first line in a script file, starting with “#!”, and it tells Linux system what interpreter to use. E.g

In the file **intron_exon_reads.py**, the first line is:

#!/usr/bin/python

* Linux is different from Windows. Windows operating system recognizes the file name extension “.py”, and rely on the name extension to decide which interpreter to use.

Make sure the binary or script file is executable:

ls -al /programs/pybedtools/bin/**intron_exon_reads.py**

-rwxr-xr-x 1 root root 3362 Mar 5 20:25 /programs/pybedtools/bin/intron_exon_reads.py

x: executable

If the file does not have the “x”, use this command to modify:
chmod uog+x /programs/pybedtools/bin/**intron_exon_reads.py**

Software installation - an overview

Many programs can be installed by simply downloading the executable file, e.g., STAR

```
wget https://github.com/alexdobin/STAR/raw/master/bin/Linux_x86_64_static/STAR
```

Static vs Shared Libraries

Linux libraries

Static

/programs/supernova-2.0.1/supernova

The executable includes all libraries. Easy to install.

Shared

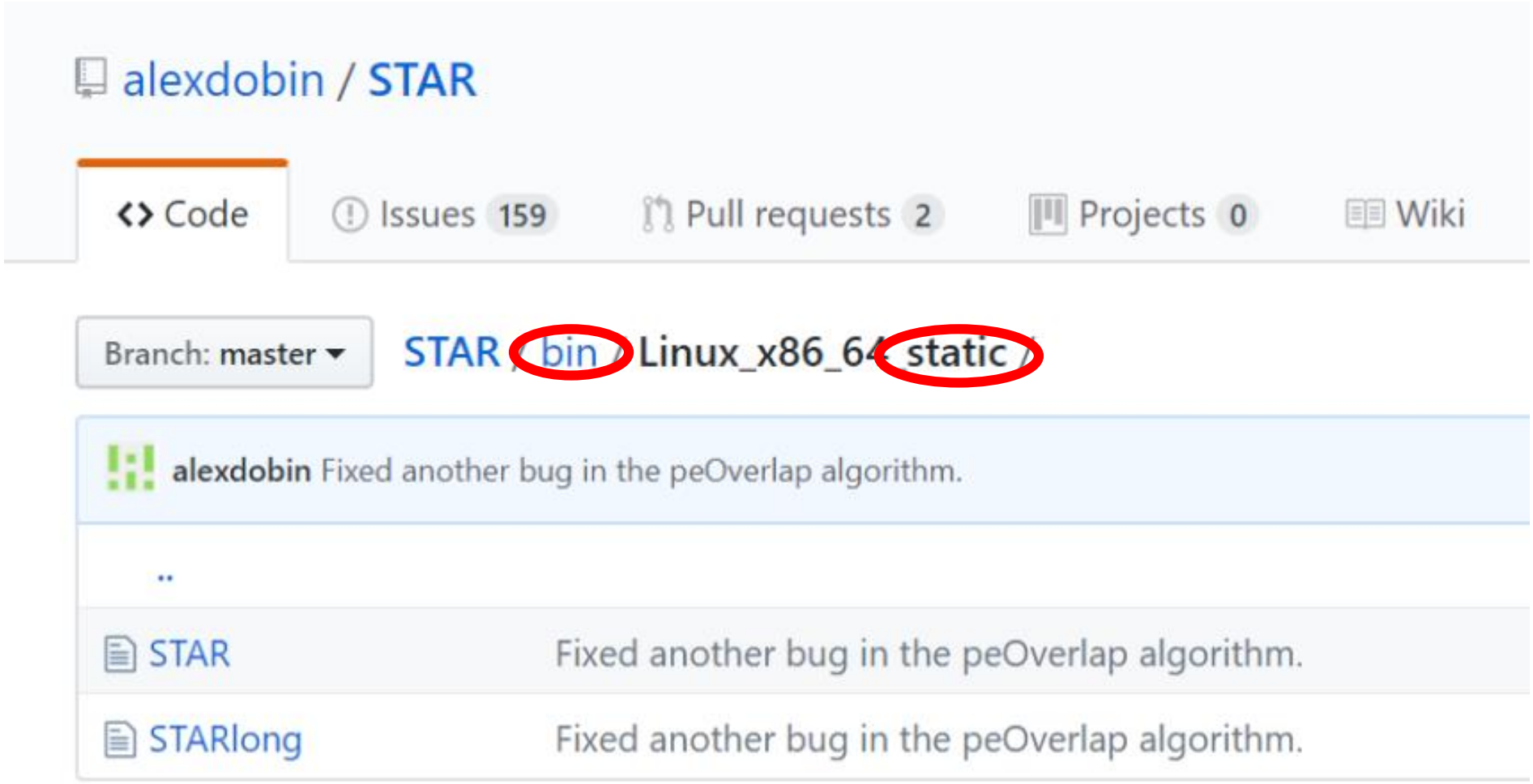
/usr/bin/gzip

/usr/lib64/libc.so.6

/usr/lib64/ld-linux-x86-64.so.2

Libraries are shared by different software. It could cause complications.

If available, first try to download the “static” “binary” version.



Make the file executable

```
ls -l STAR
-rw-rw-r-- 1 qisun qisun 8805081 May  4 08:52 STAR

chmod uog+x STAR

ls -l STAR
-rwxrwxr-x 1 qisun qisun 8805081 May  4 08:52 STAR
```

You can install multiple versions of the same software

STAR_2.3.0e.Linux_x86_64
STAR_2.4.0d
STAR_2.4.2a
STAR-2.5
STAR-2.5.2b
STAR-2.5.3a

```
export PATH=/programs/STAR-2.5/bin/Linux_x86_64_static:$PATH

which STAR

echo $PATH
```

Challenge of installing software with shared libraries:

Reproducibility

E.g. on the computer we have executable file python2.7
and library numpy-1.14.

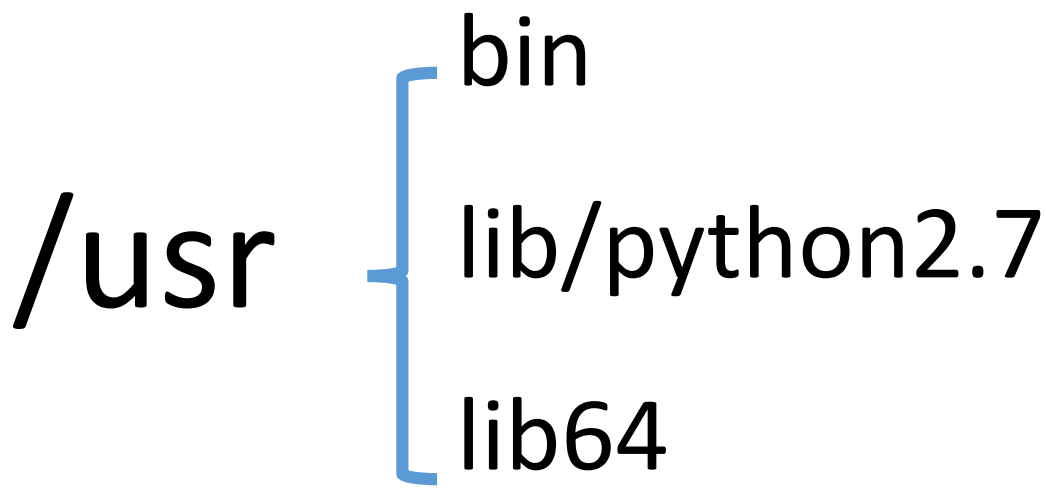
/usr
- bin/python2.7
- lib/python2.7/site-packages/numpy-1.14.3

HiCExplorer requires 15 python modules

numpy==1.13.*
scipy==1.0.*
matplotlib==2.1.*
pysam==0.11.*
intervaltree==2.1.*
…

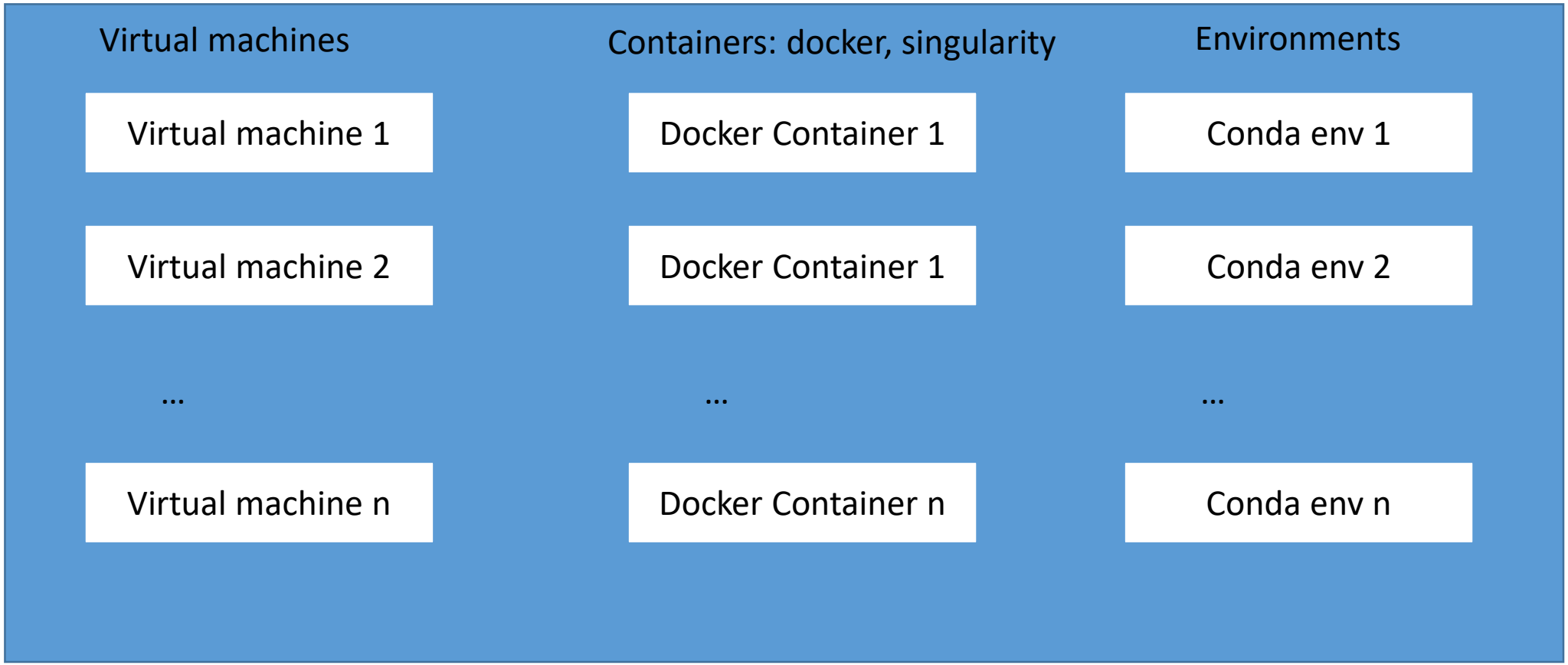
Require a different version from what on the system

If a python module inside /usr/lib is changed, software using this module could stop working, or even worse, produce different results.

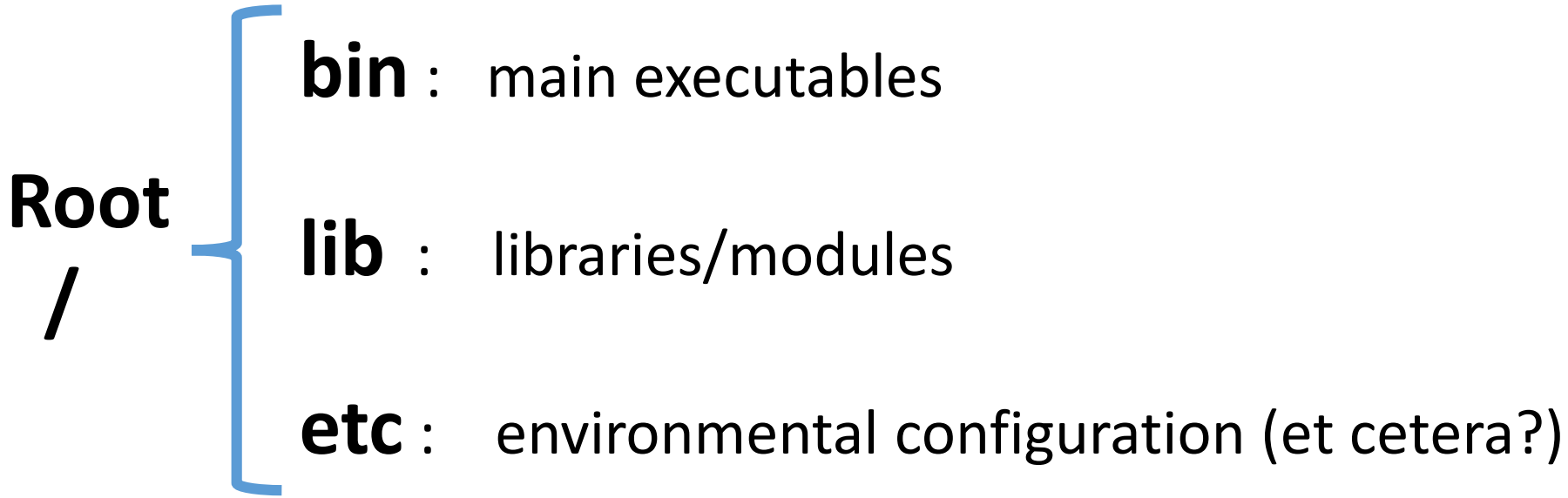


The solution: Containers and Virtual Environments

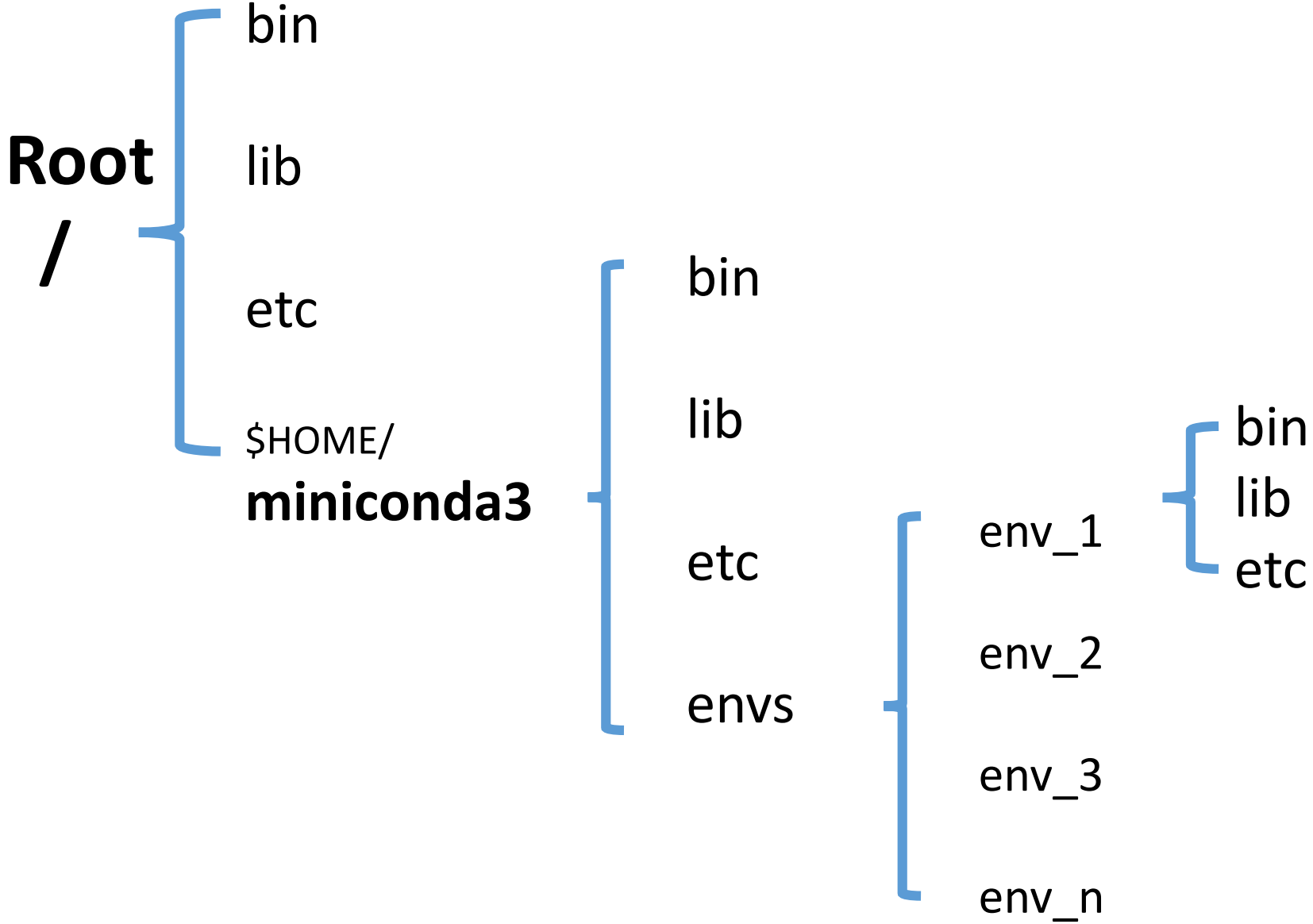
One computer can have many containers or enclosed environments. Each software is installed within a separate environment/container, with its own “eco-system”.

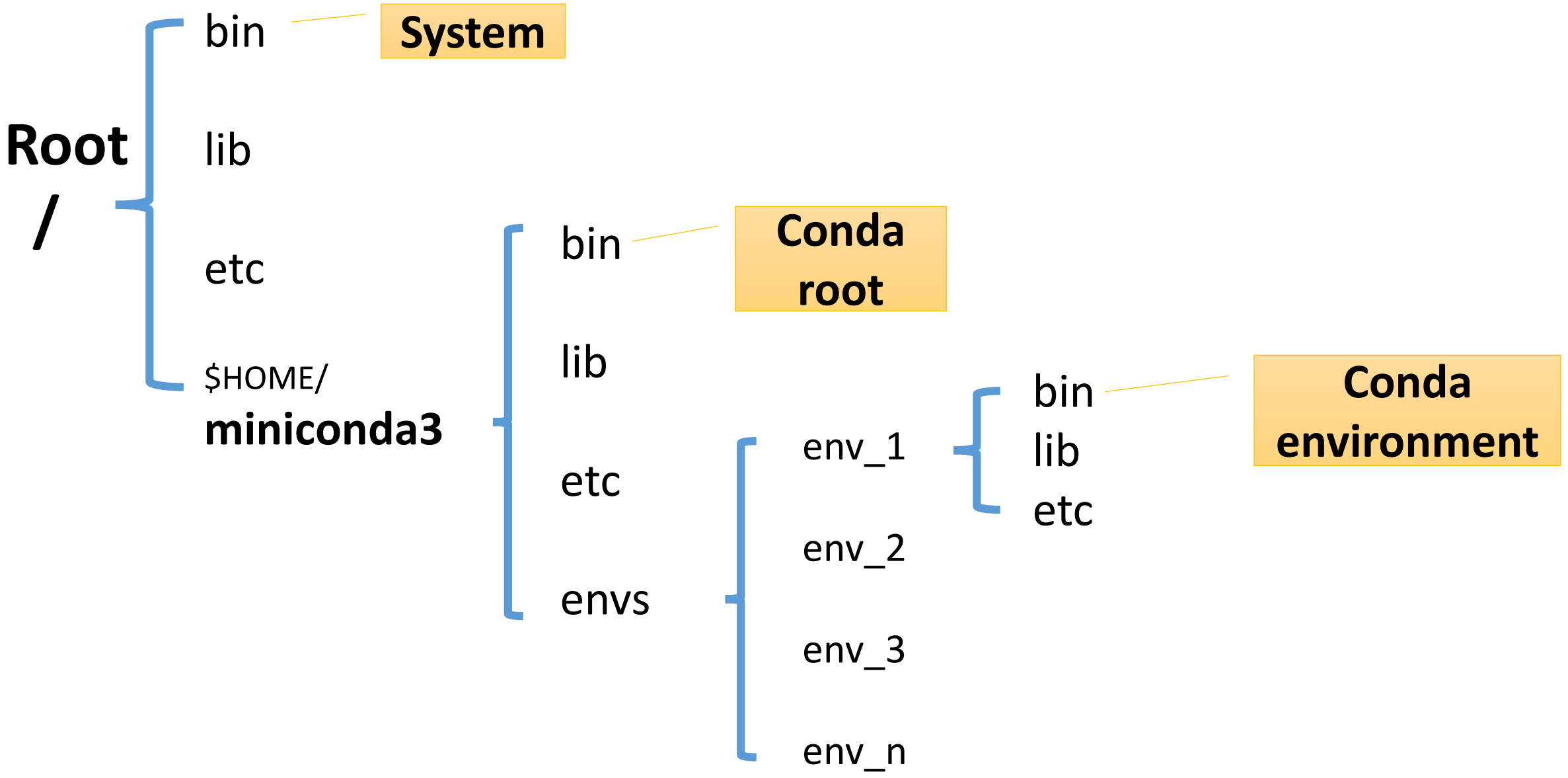


A system without Conda:

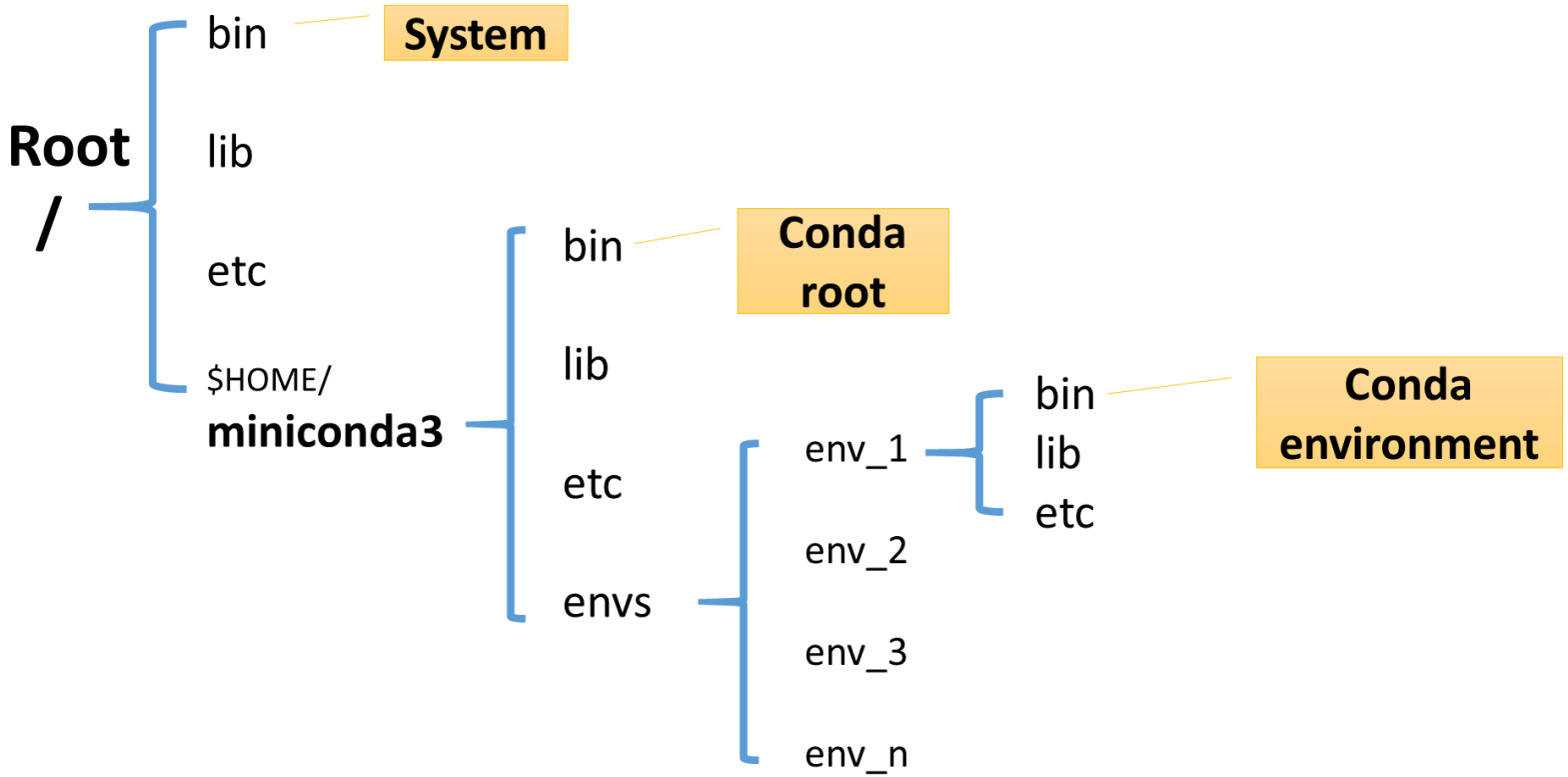


With Conda, the file system looks like this:




Root
/
bin
System
lib
etc
$HOME/
miniconda3
bin
Conda
root
lib
etc
envs
env_1
bin
lib
etc
Conda
environment
env_2
env_3
env_n

The same software can be installed at either one of the three levels.



Run BWA installed at different level:

System

```
bwa
```

Conda root

```
export PATH=$HOME/miniconda2/bin:$PATH
bwa
```

One of the Conda environments

```
export PATH=$HOME/miniconda2/bin:$PATH
source activate env_1
bwa
```

Which Conda?

Anaconda for Python2

Miniconda for Python2

Anaconda for Python3

Miniconda for Python3

- Light, no extra libraries;
- Python3 is more used now;

Where to install Conda?

- Default: home directory. (recommended)
- It can also be installed in any directories that you can write to.

How to install Conda?

https://docs.conda.io/en/latest/miniconda.html

Miniconda

	Windows	Mac OS X	Linux
Python 3.7	64-bit (exe installer)	64-bit (bash installer)	64-bit (bash installer)
	32-bit (exe installer)	64-bit (.pkg installer)	32-bit (bash installer)
Python 2.7	64-bit (exe installer)	64-bit (bash installer)	64-bit (bash installer)
	32-bit (exe installer)	64-bit (.pkg installer)	32-bit (bash installer)

Installation instructions

When installing Conda,

you will be prompted this question: Adding Conda path to .bashrc?

Please answer: **no**

(Otherwise, you won't be able to run other software we installed for you)

Do you wish the installer to prepend the miniconda3 install location to PATH in your /home/qisun/.bashrc ?
no

Install software in conda

#start conda

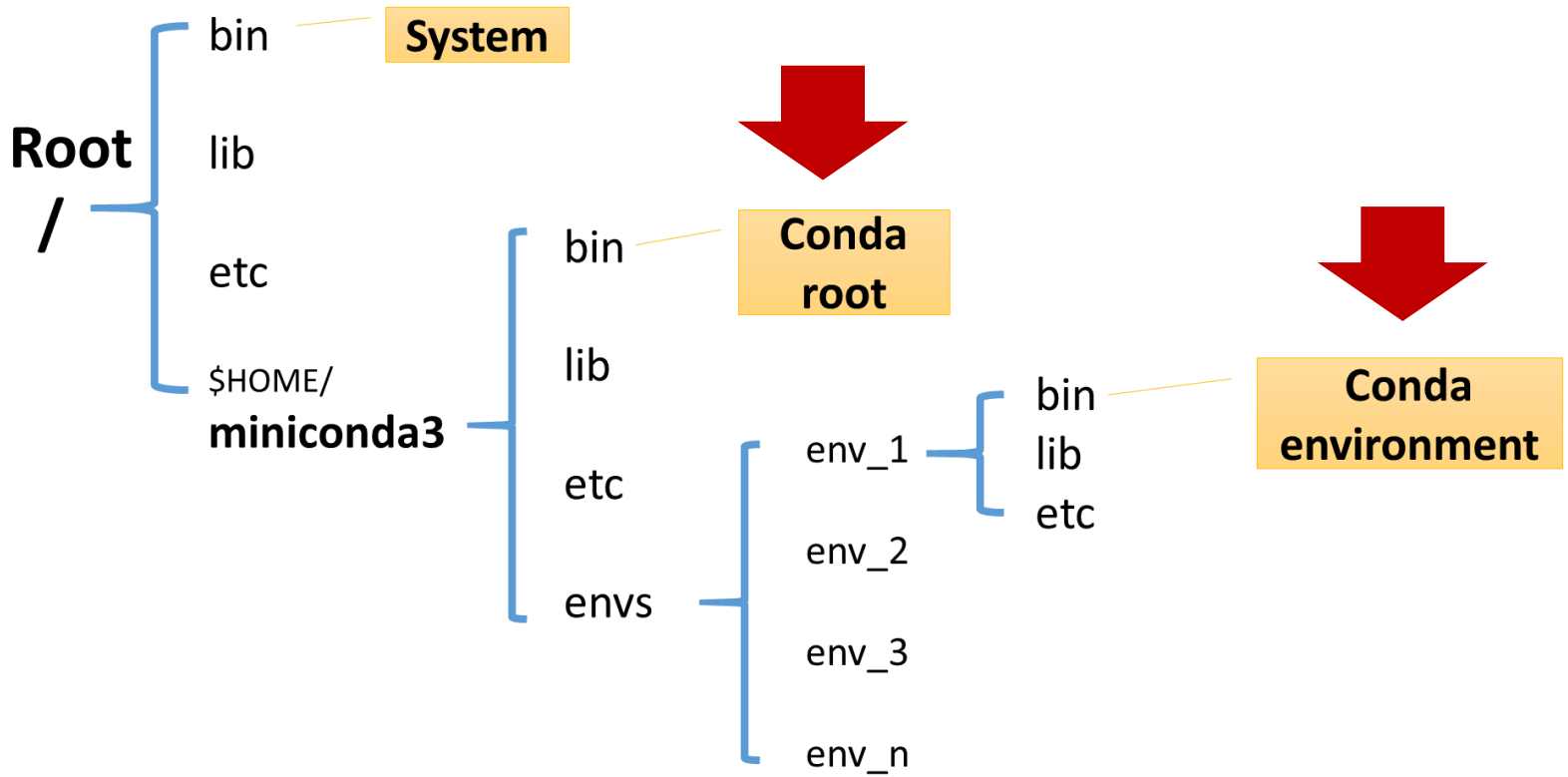
export PATH=/home/qisun/miniconda3/bin:$PATH

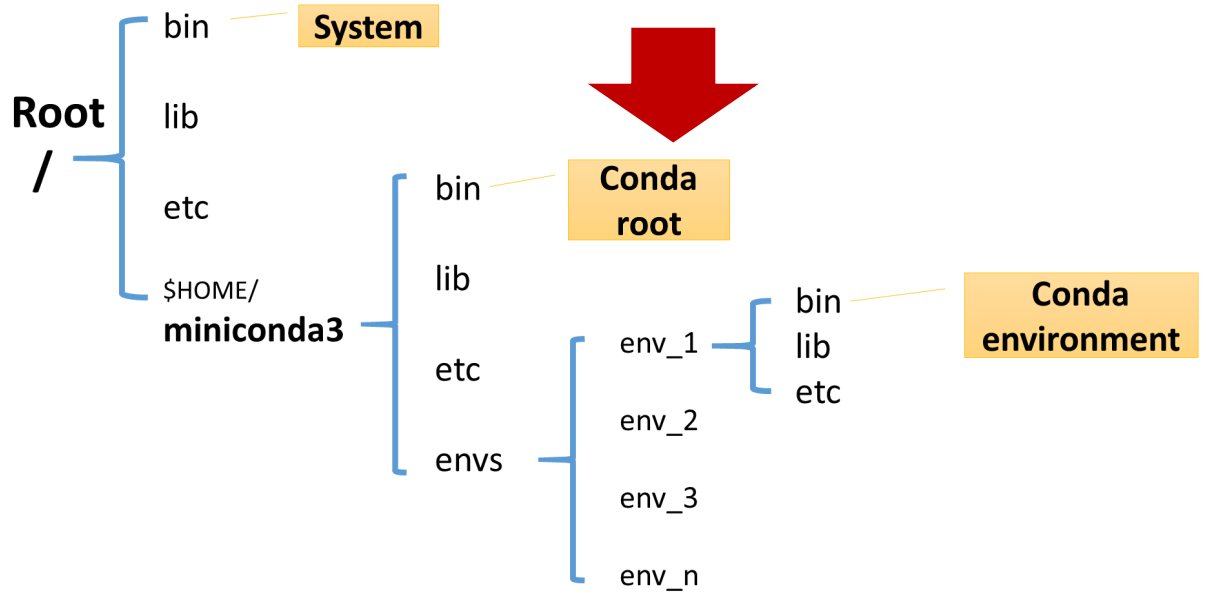
#install software

conda install blast

Install software in Conda:

You can install the same software in either of the two levels

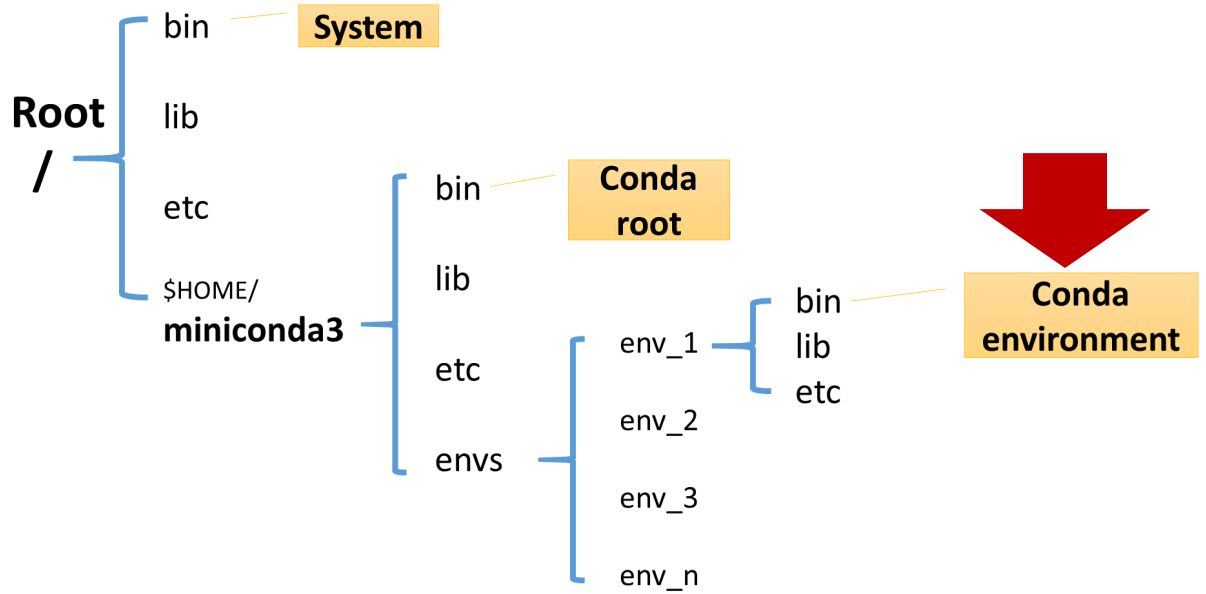




Install software in Conda root:

- If reproducibility is not critical at the stage, e.g. you are still in pipeline development phase;

- Install in Conda root is fast and save storage space, as the libraries are shared, no need to duplicate.



Install software in one of the Conda environments:

- If software is very picky about versions of libraries;

- Production pipeline, reproducibility is critical.

* An environment has its own python executable. You can have python2, python3 in different environments under the same Conda root.

The commands to install software in Conda:

Install under Conda root:

```
conda install -c bioconda blast
```

Create a Conda environment and install software:

```
conda create -c bioconda -n blast blast
```

Name of Conda channel. It is the place where conda find the package

Name of the environment you will create. It can be any name.

Name of the Conda package. This name must exists in the channel.

Create a Conda Environment without package

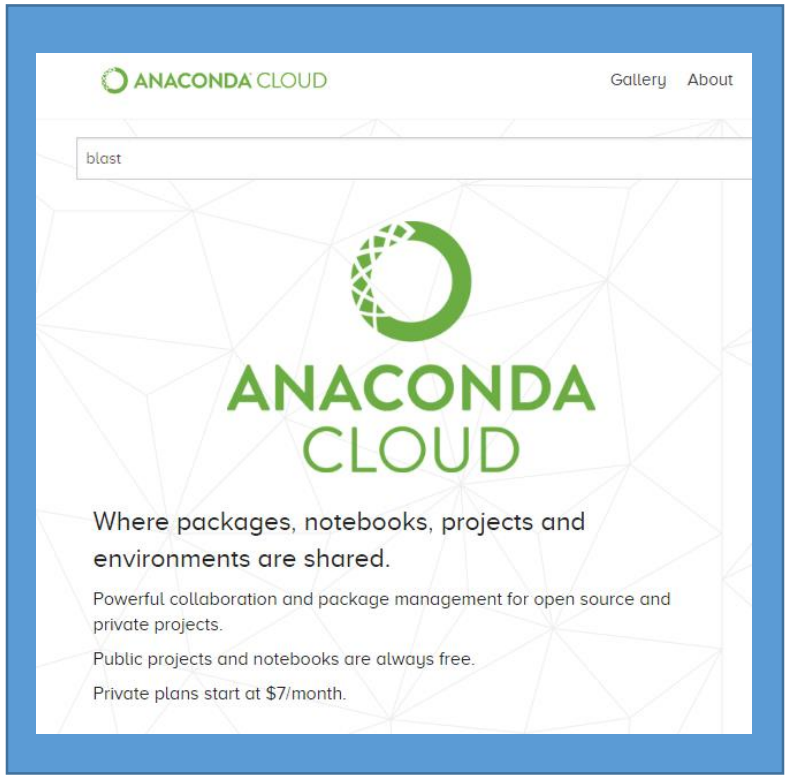
```
conda create -n myPipeLine python=3.4
```

Now, a new environment directory named “myPipeline” with python3.4.

You can use pip to install more python software

Check the package availability

https://anaconda.org/



Searching for “blast” returns:

Bioconda / blast 2.7.1

BioBuilds / blast 2.6.0

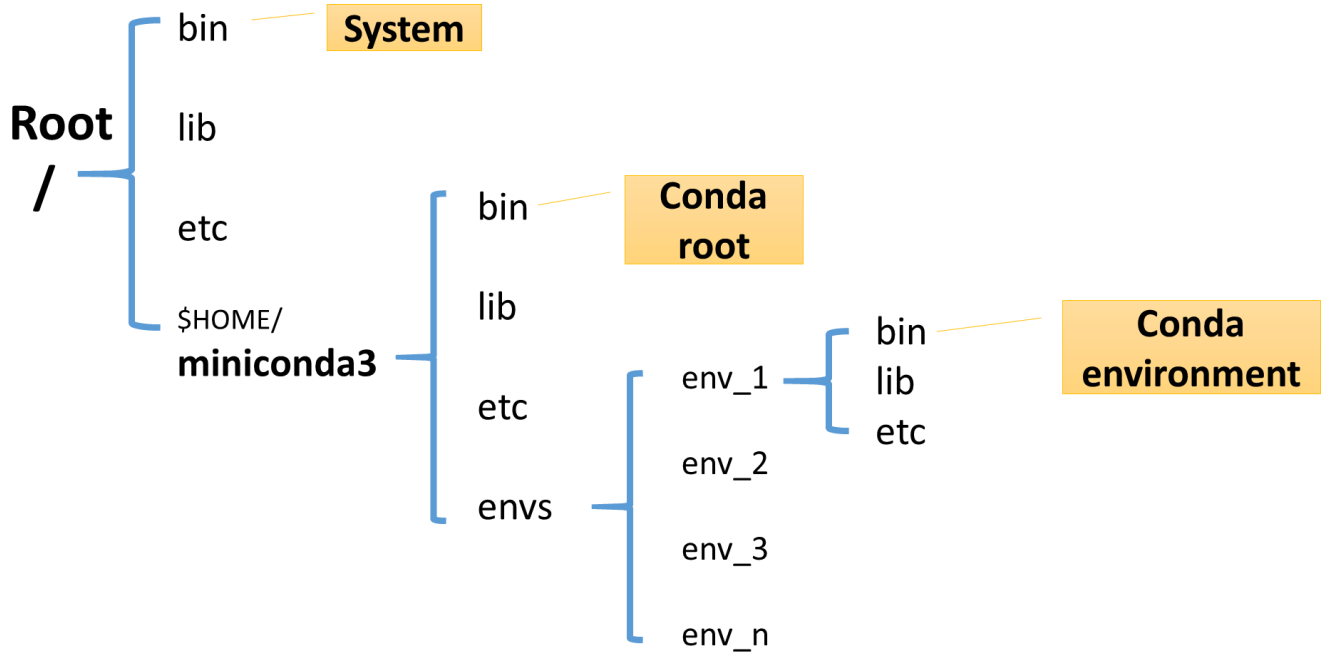
Current version at NCBI: 2.7.1

Check the software version in Conda before you use it

Use “conda update” to update an installed conda package. For example:

```
conda update biopython
```

Run software in Conda



Conda root

```
export PATH=$HOME/miniconda2/bin:$PATH
bwa
```

A Conda environment

```
export PATH=$HOME/miniconda2/bin:$PATH
source activate env_1
bwa
```

How does Conda work?

Starting Conda:

export PATH=/home/qisun/miniconda3/bin:$PATH

What is inside **/home/qisun/miniconda/bin:**

python
pip
…

You are using this copy of python, which manages its own libraries

Run software in Conda environment

```
# Start Conda environment, add environment directory to the $PATH:
source activate myPipeLine

# Run software
mySoftware

# Stop Conda virtual environment
conda deactivate
```

A few more things:

Once you are in Conda, you can full privilege to install anything.

You can install other python modules either with pip or conda to install more python modules. But using “conda install” whenever possible.

```
pip install numpy

conda install numpy
```

Avoid setting PYTHONPATH and LD_LIBRARY_PATH when using Conda.

If needed, clear PYTHONPATH and LD_LIBRARY_PATH if they could interfere with Conda.

```
unset PYTHONPATH

unset LD_LIBRARY_PATH
```