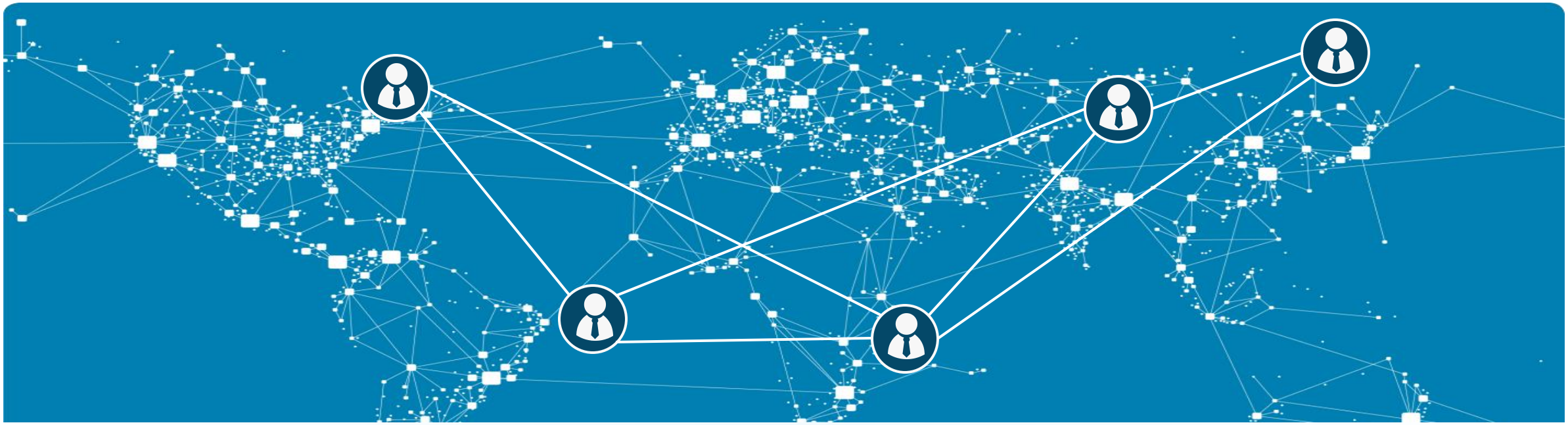
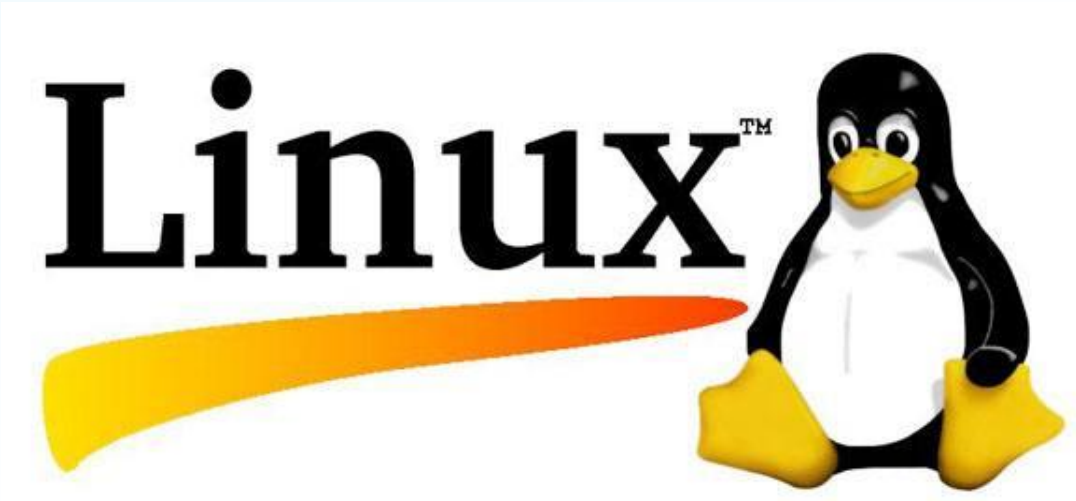
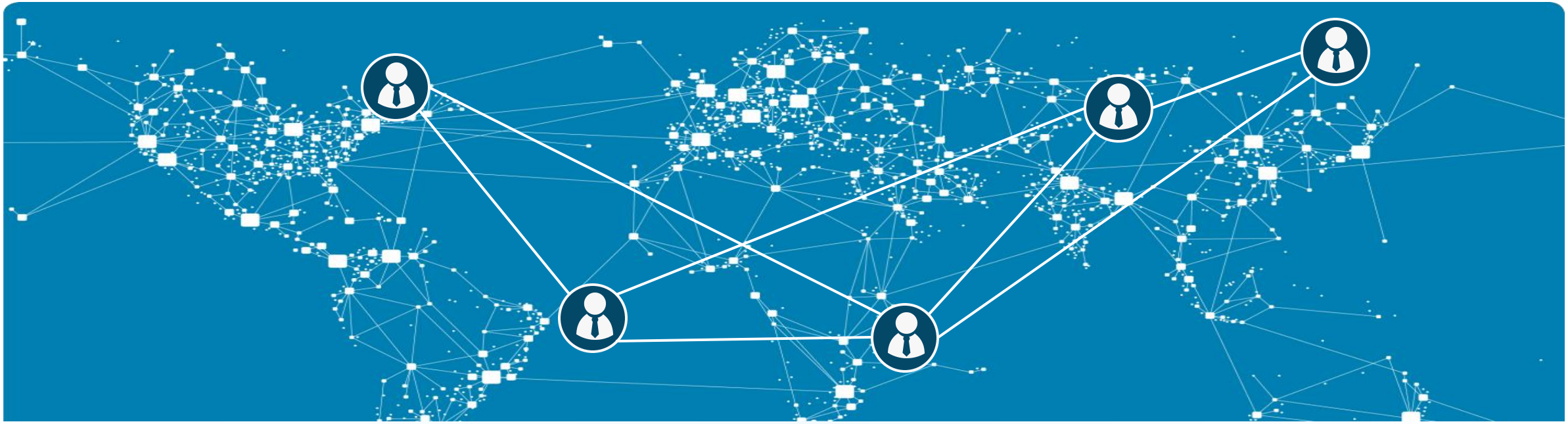


12-13Linux软件安装



Linux服务器配置的软件

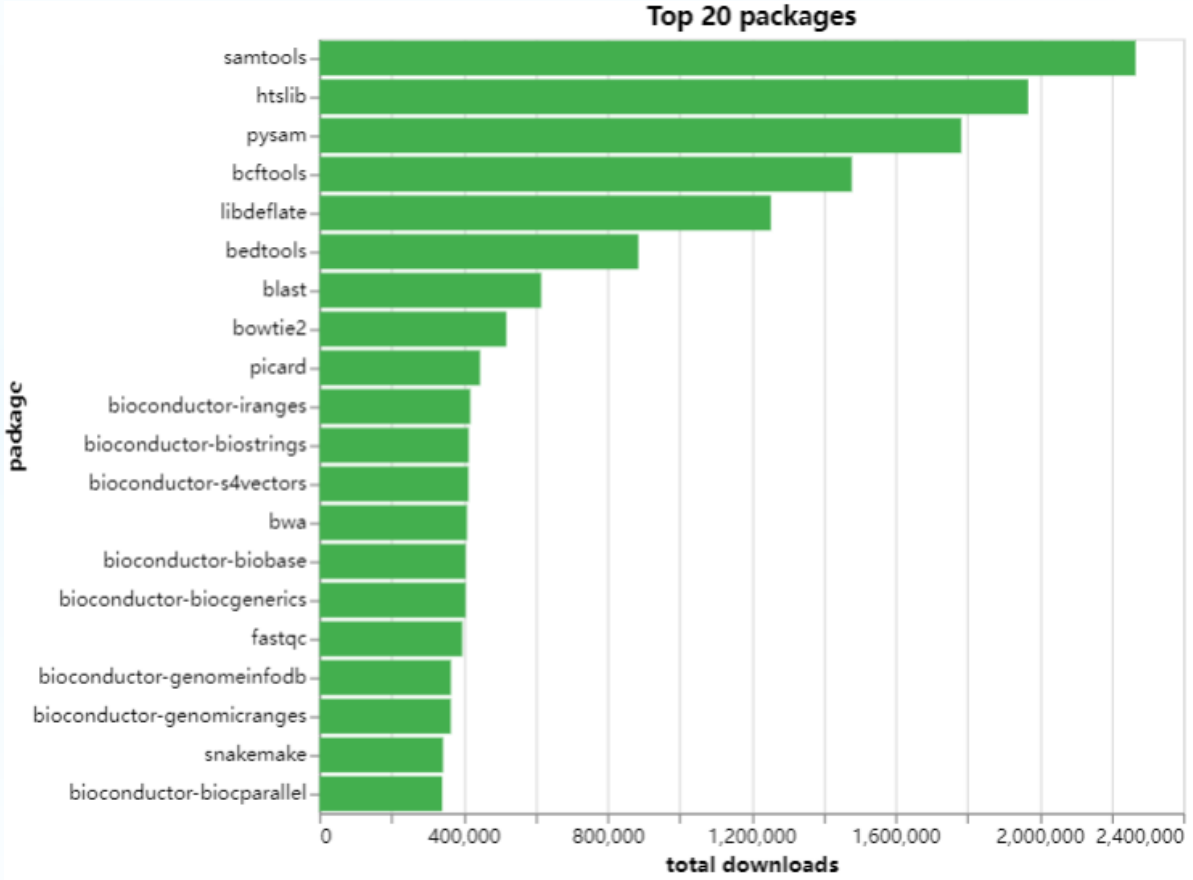
- 质控程序：fastqc，multiqc，kneaddata
- 基于Reads的分析：HUMAnN2, GraPhlAn, Kraken2
- 基于Contigs的分析：megahit, spades, quast, prokka, cd-hit, emboss, salmon, eggnog
- 分箱工具：metawrap, checkm, drep, gtdb
- 小工具：rush csvtk seqkit



Conda安装

Conda软件包管理神器

- conda：任意语言的软件包、环境、依赖关系的开源管理系统。
- Anaconda：集合了常用Python包的数据科学平台
- Miniconda：精简版Anaconda，只包含conda和Python
- bioconda：conda的一个通道，含数万生信分析软件和版本收录，文章发表于Nature Method



[一文掌握Conda软件安装：虚拟环境、软件通道、加速solving、跨服务器迁移]

(不) 推荐使用miniconda，可以获得最新版

- wget -c https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86_64.sh
- # -p 指定安装路径
- # -b -f 不做提示，直接安装
- bash Miniconda3-latest-Linux-x86_64.sh -b -f -p ${HOME}/miniconda3

Conda增加国内通道（可选）

- conda config --add channels https://mirrors.tuna.tsinghua.edu.cn/anaconda/cloud/msys2/# Anocanda清华镜像，国内镜像，加速下载
- conda config --add channels https://mirrors.tuna.tsinghua.edu.cn/anaconda/pkgs/free/
- conda config --add channels https://mirrors.tuna.tsinghua.edu.cn/anaconda/pkgs/main/
- conda config --add channels https://mirrors.tuna.tsinghua.edu.cn/anaconda/cloud/bioconda/
- conda config --set show_channel_urls yes

Conda增加通道

- conda config --add channels defaults
- conda config --add channels bioconda # 增加软件支持
- conda config --add channels conda-forge # Highest priority

- conda config --set show_channel_urls yes

一文掌握Conda软件安装：虚拟环境、软件通道、加速solving、跨服务器迁移

推荐使用mamba，可以获得最新版

- curl -L -O "https://github.com/conda-forge/miniforge/releases/latest/download/Miniforge3-$(uname)-$(uname -m).sh"
- # 因为历史原因，一直把 conda 安装在了/anaconda3下面
- bash Miniforge3-$(uname)-$(uname -m).sh -b -p /anaconda3
- /anaconda3/bin/mamba init bash
- source ~/.bashrc

一文掌握Conda软件安装：虚拟环境、软件通道、加速solving、跨服务器迁移

创建新的运行环境

- 创建meta环境，指定使用的R和python版本

  mamba create -y -n meta

- 激活新环境meta

  source activate meta

  mamba activate meta

- 退出meta环境

  source deactivate meta

  mamba deactivate meta

激活环境、按顺序安装软件

- 在新环境meta中安装软件

```
# 先激活环境，如下运行
# conda install mamba
mamba install rush -c bioconda
mamba install csvtk -c bioconda
mamba install seqkit -c bioconda
mamba install kneaddata=0.7.4 -c bioconda –y
mamba install humann2=2.8.1
```

新版Mamba体验超快的软件安装 https://mp.weixin.qq.com/s/6AI1nGfSHtDl-WWhHNtWyg

Conda很好用，但需要联网

- 安装检测冲突慢
- Conda软件下载慢
- 单位服务器不能联网

一文掌握Conda软件安装：虚拟环境、软件通道、加速solving、跨服务器迁移

Conda-pack助力快速复制环境

- Conda pack 直接解压我们打包好的环境
- 拷贝 – 解压 – 激活 即可使用
- 无需下载，无需联网，速度快，成功率100%

一文掌握Conda软件安装：虚拟环境、软件通道、加速solving、跨服务器迁移

Conda-pack 打包环境

- 安装好的环境打包导出，以宏基因组kraken2为例

```
# conda环境包统一存放
# 设置环境名，如metagenome_env meta humann2
n=kraken2
conda pack -n ${n} -o ${n}.tar.gz
# 导出文件列表
conda activate ${n}
conda env export > ${n}.yml
```

Conda-unpack 解包环境

- 打包好的环境在新服务器解包，以宏基因组kraken2为例

```
# 指定环境名称，如meta，  metawrap1.3, humann2, humann3, qiime2-2021.2
n=meta
wget -c http://210.75.224.110/db/conda/${n}.tar.gz     # 下载
mkdir -p ~/miniconda3/envs/${n} # 指定安装目录
tar -xvzf ${n}.tar.gz -C ~/miniconda3/envs/${n}
source ~/miniconda3/envs/${n}/bin/activate # 激活环境
conda unpack     #解包
```

Conda pack直接加载已经安装好的环境 (完整版)

- # 下载conda
  wget -c https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86_64.sh
- # 安装
  bash Miniconda3-latest-Linux-x86_64.sh -b –f
- # 初始化
  ~/miniconda3/condabin/conda init
  source ~/.bashrc
- # 下载已安装好的环境 metagenome_env.tar.gz
  tar –xzvf metagenome_env.tar.gz -C ~/miniconda3/envs/metagenome_env
- # 激活环境
  source ~/miniconda3/envs/metagenome_env/bin/activate
  conda-unpack

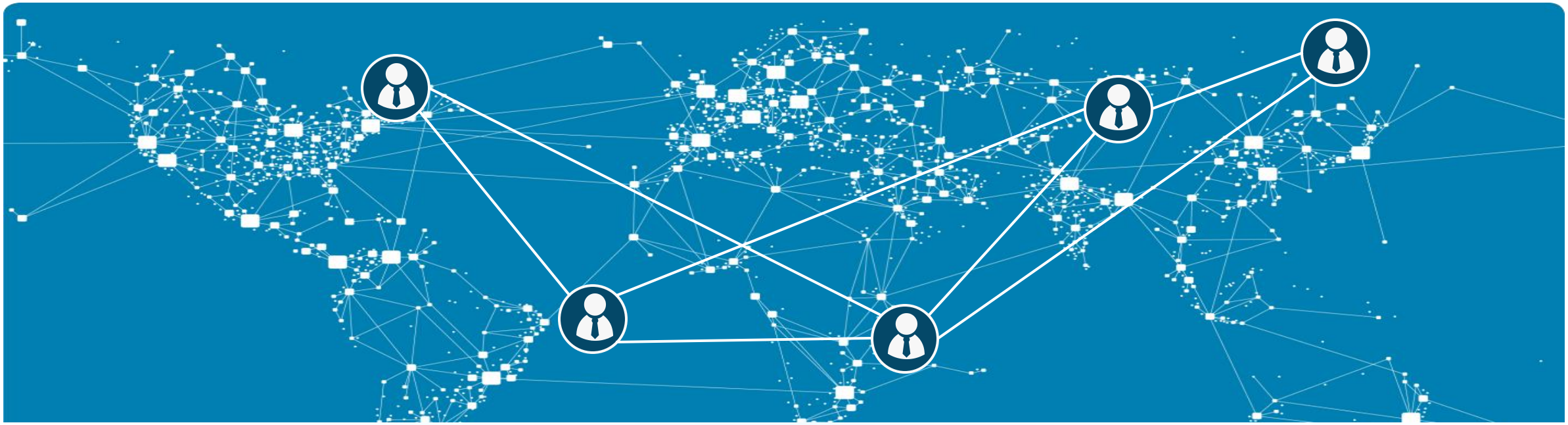
Conda 环境激活和去激活

```
n=kraken2
# 方法1. 简单激活环境
conda activate $n
conda activate kraken2
# 退出环境
conda deactivate
# 方法2. 全路径激活环境，适用范围更广，但麻烦
source ~/miniconda2/envs/${n}/bin/activate
```

判断可执行文件的位置和当前所在环境

- which python返回python的位置
- 激活conda的环境在终端会有标识

```
(metagenome_env) amplicon@localhost:~$ which python
/anaconda2/envs/metagenome_env/bin/python
(metagenome_env) amplicon@localhost:~$ source deactivate metagenome_env
amplicon@localhost:~$ which python
/usr/bin/python
amplicon@localhost:~$ source activate metagenome_env
(metagenome_env) amplicon@localhost:~$
```



几个需要注意的概念（略过）

Linux下的软件运行所需的3个条件

- 软件类型
  - 脚本
  - 二进制程序
  - Java包
- 可执行属性
  - 软件或脚本需要有执行权限 `chmod a+x soft_name`
- 环境变量
  - 告诉系统软件可能在的位置或使用完整路径

什么样的软件?

- **脚本型**

  解释型语言写作，如Bash，R，Python，Perl等，源代码可直接查看

- **二进制型**

  源代码编译成机器语言，直接打开查看乱码，如bwa, salmon等

- **Java程序**

  java -jar trimmomatic-0.36.jar

```
(metagenome_env) amplicon@localhost:~$ head `which kraken`
#!/usr/bin/env perl

# Copyright 2013-2015, Derrick Wood <dwood@cs.jhu.edu>
#
# This file is part of the Kraken taxonomic sequence classification system.
#
# Kraken is free software: you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# (at your option) any later version.
```

```
(metagenome_env) amplicon@localhost:~$ head `which salmon`
LF>8P@H/u@8     @*'@@@@88@8@@@
 "¤NUGNUAr
"*A
44!

,ª@
P       Б`JЄDA @B`!"(P¼,¬1 d
<Ð $@@"! 4$eR¤5*T!BC ) ` 0in~@ !@DiP¢J,N      3$0p -X@.AD
```

可执行属性 – 软件的必须属性

- ls –l 查看文件的属性 （文件夹的可执行属性是可读属性）
- 一般在终端不同颜色对应不同属性
- chmod修改属性
- chmod a+x file # 所有人增加可执行属性
- chmod 755 file # 所有人可执行，自己可写

```
drwxr-xr-x  4 ct ct    32 10月 14 10:21 binning
-rw-r--r--  1 ct ct 15702 10月 18 16:33 metagenome_softinstall_ln_wgt.sh
-rwxr-xr-x  1 ct ct 19272 10月 14 10:21 pipeline.sh
drwxr-xr-x 12 ct ct   190 10月 14 10:21 result
drwxr-xr-x  2 ct ct   258 10月 14 10:21 seq
-rw-r--r--  1 ct ct  5804 10月 14 10:21 soft_db.sh
drwx------  2 ct ct     6 10月 18 13:48 temp
lrwxrwxrwx  1 ct ct     4 10月 18 20:24 temp2 -> temp
```

环境变量PATH – 软件所在目录的集合

- 环境变量PATH是一堆目录，一堆**存放有软件的目录。**
- 在系统接到命令输入比如”cd”后，会去环境变量PATH存储的目录中从前向后查找，在哪个目录发现存在输入的命令同名”cd”的文件视为找到程序，然后判断是否有可执行属性，如果有则执行。
- echo $PATH
- export PATH=$PATH:~/soft

```
ct@localhost:/db/meta$ echo $PATH
/self_bin:/disk2/bin:/anaconda2/bin/:/usr/lib64/qt-3.3/bin:/disk2/home/ct/perl5/bin:/usr/local/bin:/usr/bi
n:/usr/local/sbin:/usr/sbin:/disk2/bin/:/anaconda2/bin:/opt/Cytoscape_v3.5.1/:/disk2/soft/rsem/bin:/disk2/
soft/bin:/disk2/home/ct/bin
ct@localhost:/db/meta$ export PATH=$PATH:~/soft
ct@localhost:/db/meta$ echo $PATH
/self_bin:/disk2/bin:/anaconda2/bin/:/usr/lib64/qt-3.3/bin:/disk2/home/ct/perl5/bin:/usr/local/bin:/usr/bi
n:/usr/local/sbin:/usr/sbin:/disk2/bin/:/anaconda2/bin:/opt/Cytoscape_v3.5.1/:/disk2/soft/rsem/bin:/disk2/
soft/bin:/disk2/home/ct/bin:/disk2/home/ct/soft
```

环境变量

PATH和path，傻傻分不清

```
YSX@ehbio:~/train/single_cell$ pipeline_metagenome.sh
-bash: pipeline_metagenome.sh: 未找到命令
```

`pipeline_metagenome.sh` 命令去哪儿了？上面我们都看到了，就在 `metagenome` 目录下，为啥电脑（操作系统）这么笨却找不到？另外为什么运行 `head` 就可以找到？难道有一些黑魔法在里面？

确实是有一些黑魔法的，不过我们一般称之为**规则**。

操作系统为了便捷性和安全性，定义了一系列环境变量，存储常用信息，`PATH` (注意全是大写)是其中一个。

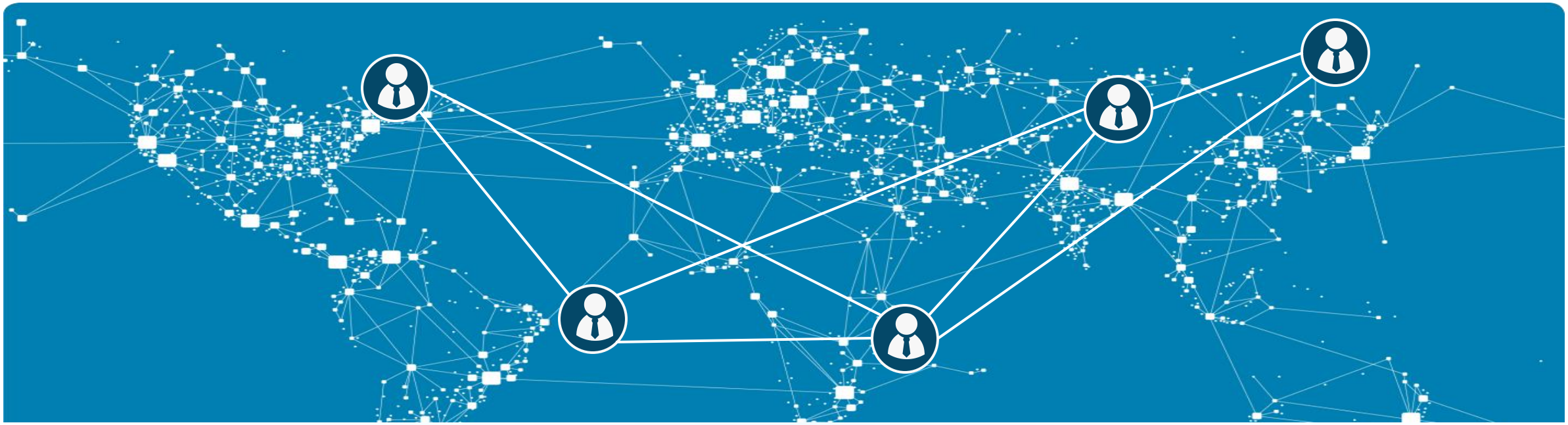
`PATH`：是存放有(可执行)命令和程序的目录集合；在操作系统接到用户输入的命令时，会对PATH**存储的目录**进行查找，看下是否有与用户输入的命令同名的文件存在，而且是**从前到后**一个个查找，而且是**查到就停**，最后查不到就报错。（从这几个**加粗**的文字，可以看到操作系统很懒，当然懒是好的程序员的必备属性。）

环境变量 – 永久设置

- 服务器自己用户的家目录下一般有2个隐藏文件： .bashrc和.bash_profile。
- .bashrc本地登录时读取。
- .bash_profile远程登录时读取。
- .bashrc和.bash_profile是bash脚本，可以写任何bash命令。
- 需要把环境变量设置命令写入.bash_profile中。

不同类型的“环境变量”

- 环境变量PATH：定义可执行程序的目录
- LD_LIBRARY_PATH：定义动态库的目录 (.so文件not found)
- PYTHONPATH：定义Python包的目录
- PERL5LIB：定义Perl模块的目录



传统软件安装方法

编译好的二进制文件

- 编译好的多平台通用二进制文件或特定平台可用二进制文件，下载，解压，增加可执行属性，放入环境变量，直接调用。
- 认真看软件说明手册，如果提供了二进制版本，尽量使用二进制版本，简单方便，把时间多放在数据上，而不是软件安装上。
- 一般可执行程序放置在 bin 目录下。

```
wget ftp://ftp.ncbi.nlm.nih.gov/blast/executables/blast+/LATEST/ncbi-blast-2.7.1+-x64-lir
tar xvzf ncbi-blast-2.7.1+-x64-linux.tar.gz
cd ncbi*
cd bin
# 直接进入 bin 目录，找到对应可执行文件，链接到在环境变量的目录中去。
# 具体可看视频的操作  http://bioinfo.ke.qq.com
ln -s `pwd`/* ~/bin
```

经典的源码安装

- ./configure && make && make install
- ./configure是检测系统的库文件、头文件、依赖的软件是否存在及版本是否兼容，并根据检测结果生成Makefile文件。这一步是软件安装是否能成功的关键，检测通过安装一般没问题。检测不通过，缺什么补什么。如果非根用户，这一步通常也会配置下软件安装的路径--prefix=/home/ct/soft/specific_name。
- make具体的编译过程，根据Makefile中的规则把程序语言转换为机器语言。
- make install拷贝make编译出的可执行文件或依赖的动态库到prefix指定的目录。
- 置入环境变量即可使用。

```
wget https://jaist.dl.sourceforge.net/project/samtools/samtools/1.7/samtools-1.7.tar.bz2
tar xvzf samtools-1.7.tar.bz2
cd samtoo*
./configure --prefix=/home/ct/soft/samtools
make
make install
cd /home/ct/soft/samtools/bin
ln -s `pwd`/* ~/bin
```

Python包的安装

- Python包管理器安装

  easy_install package_name

  pip install package_name -i https://pypi.tuna.tsinghua.edu.cn/simple/

- Python包手动源码安装

  python setup.py build

  python setup.py install

- Conda安装

  conda install package_name

```
(eggnog-mapper) [root@localhost eggnog]# python --version
Python 3.9.7
(eggnog-mapper) [root@localhost eggnog]# python summarizeAbundance.py -i FattyAc
id/result/salmon/gene.TPM -m FattyAcid/temp/eggnog/output.emapper.annotations -c
 '7,12,19' -s '*+,+,' -n raw -o FattyAcid/result/eggnog/eggnog
Traceback (most recent call last):
  File "/home/seesea/Desktop/FattyAcid/result/eggnog/summarizeAbundance.py", lin
e 45, in <module>
    import pandas as pd
ModuleNotFoundError: No module named 'pandas'
(eggnog-mapper) [root@localhost eggnog]#
```

软件和数据库下载

- wget -c soft_url/database_url # -c断点续传
- wget -c ftp://ftp.ncbi.nlm.nih.gov/blast/db/nt.*.tar.gz 支持通配符
- wget -cr -np -nd ftp://ftp.ncbi.nlm.nih.gov/blast/db/ 递归下载
- 同类工具还有curl和axel，都可以用

数据备份 rsync

- rsync则是一个增量备份工具，只针对修改过的文件的修改过的部分进行同步备份，大大缩短了传输的文件的数量和传输时间。

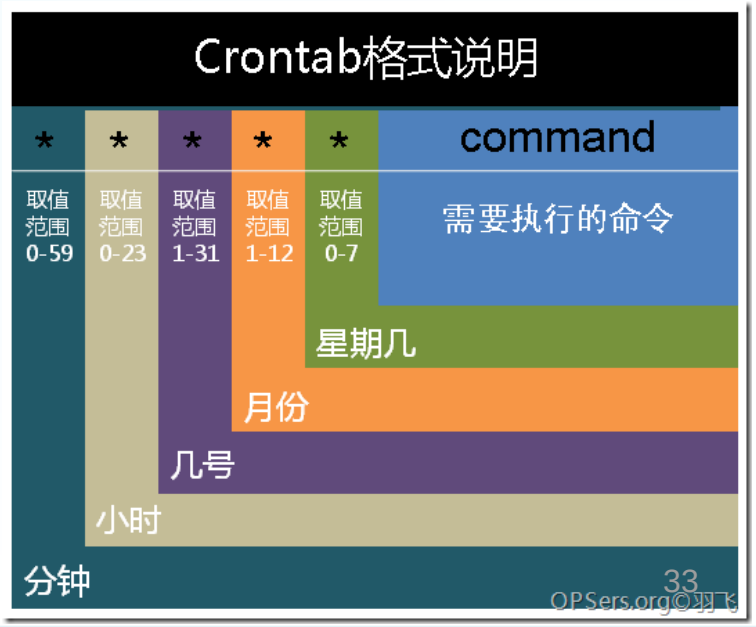
```
# 把本地project目录下的东西备份到远程服务器的/backup/project目录下
# 注意第一个project后面的反斜线，表示拷贝目录内的内容，不在目标目录新建project文件夹。
# -a: archive mode, quals -rlptgoD
# -r: 递归同步 -p: 同步时保留原文件的权限设置
# -u: 若文件在远端做过更新，则不同步，避免覆盖远端的修改
# -L: 同步符号链接链接的文件，防止在远程服务器出现文件路径等不匹配导致的软连接失效
# -t: 保留修改时间 -v: 显示更新信息 -z: 传输过程中压缩文件，对于传输速度慢时适用
rsync -aruLptvz --delete project/ user@remoteServer:/backup/project
```

https://mp.weixin.qq.com/s/c2cspK5b4sQScWYMBtG63g

定时任务

- crontab -e # 打开下面的编辑vim界面

```
#minute hour    day     month   week    command
0       0       */3     *       *       rsync.sh at 00:00 every 3 days
0       6       *       *       *       Call me at 6:00 everyday
*/20    6-18    *       *       *       Run every 20 minutes in 6-18
```



命令找不到？权限不够？

- 程序名字有没有拼错？
- 软件有没有安装？
- 安装好的软件有没有在环境变量里面？
  - 未设置环境变量
  - 环境变量设置后未更新
- 软件有没有可执行权限？

```
metaphlan_to_stamp.pl result/metaphlan2/taxonomy.tsv \
>       > result/metaphlan2/taxonomy.spf
-bash: metaphlan_to_stamp.pl: command not found
(metagenome_env_ehbio) zkzhou@iascaas 11:46:29
 ~/zryuan/meta_ehbio_testdata
```

```
(metagenome_env) root@localhost:/tmp# a
-bash: /conda2/envs/metagenome_env/bin/a: 权限不够
(metagenome_env) root@localhost:/tmp# ls -l /conda2/envs/metagenome_env/bin/a
-rw-r--r-- 1 root root 0 1月  20 19:00 /conda2/envs/metagenome_env/bin/a
```

```
meta0724_zhoulei@localhost:~/meta/binning$  conda activate metawrap
bash: conda: 未找到命令...
```

如果出现conda core dump怎么解决?

```
(meta) ug0275@genek:~/db$     conda install -c cyclus java-jdk # 156M, cyclus极慢
Collecting package metadata (current_repodata.json): done
Solving environment: done
Segmentation fault (core dumped)
(meta) ug0275@genek:~/db$
```

```
(meta) ug0275@genek:~/db$     conda install kneaddata # kneaddata v0.7.4,
Collecting package metadata (current_repodata.json): done
Solving environment: done
Segmentation fault (core dumped)
(meta) ug0275@genek:~/db$     kneaddata --versi
kneaddata: command not found
(meta) ug0275@genek:~/db$
```

Conda软件安装 core dump error/Segment fault/段错误怎么办

```
# 清空缓存
# https://github.com/conda/conda/issues/7815
conda clean -a
```

```
(base) [sc30177@ln1:~] $ conda install -c biobakery humann
Collecting package metadata (current_repodata.json): done
Solving environment: failed with initial frozen solve. Retrying with flexible solve.
Solving environment: failed with repodata from current_repodata.json, will retry with next
repodata source.
Collecting package metadata (repodata.json): done
Solving environment: failed with initial frozen solve. Retrying with flexible solve.
Solving environment: -
Found conflicts! Looking for incompatible packages.
This can take several minutes.  Press CTRL-C to abort.
failed

UnsatisfiableError: The following specifications were found to be incompatible with each ot
her:

Output in format: Requested package -> Available versions
```

IO错误一般是权限问题

```
(eggnog-mapper=2.0.1) [zryuan@mu01 meta_ehbio_testdata]$ time emapper.py --annotate_hits_table \
>       temp/eggnog/protein.emapper.seed_orthologs --no_file_comments \
>     -o temp/eggnog/output --cpu 16 --data_dir ${db}/eggnog2 --override
/home/zryuan/anaconda2/envs/eggnog-mapper=2.0.1/bin/diamond /home/zryuan/anaconda2/envs/eggnog-mapper=2.0.1/
lib/python2.7/site-packages
#  emapper-2.0.1
# ./emapper.py  --annotate_hits_table temp/eggnog/protein.emapper.seed_orthologs --no_file_comments -o temp/
eggnog/output --cpu 16 --data_dir /lustre/zryuan/db/eggnog2 --override
Traceback (most recent call last):
  File "/home/zryuan/anaconda2/envs/eggnog-mapper=2.0.1/bin/emapper.py", line 1213, in <module>
    main(args)
  File "/home/zryuan/anaconda2/envs/eggnog-mapper=2.0.1/bin/emapper.py", line 250, in main
    annotate_hits_file(args.annotate_hits_table, annot_file, hmm_hits_file, args)
  File "/home/zryuan/anaconda2/envs/eggnog-mapper=2.0.1/bin/emapper.py", line 714, in annotate_hits_file
    seq2annotOG = annota.get_ogs_annotations(set([v[0] for v in seq2bestOG.itervalues()]))
  File "/home/zryuan/anaconda2/envs/eggnog-mapper=2.0.1/lib/python2.7/site-packages/eggnogmapper/annota.py",
 line 34, in get_ogs_annotations
    if db.execute(cmd):
sqlite3.OperationalError: disk I/O error
```

IO错误一般是权限问题

```
21  def get_ogs_annotations(ognames):
22      # og VARCHAR(16) PRIMARY KEY,
23      # level VARCHAR(16),
24      # nm INTEGER,
25      # description TEXT,
26      # COG_categories VARCHAR(8),
27      # GO_freq TEXT,
28      # KEGG_freq TEXT,
29      # SMART_freq TEXT,
30      # proteins TEXT);
31      query = ','.join(map(lambda x: '"%s"'%x, ognames))
32      cmd = 'SELECT og.og, description, COG_categories FROM og WHERE og.og IN (%s)' % query
33      og2desc = {}
34      if db.execute(cmd):
35          for og, desc, cat in db.fetchall():
36              cat = re.sub(cog_cat_cleaner, '', cat)
37              og2desc[og] = [cat, desc]
38      return og2desc
39
```

Core dump是软件内存问题，成熟的软件发生这个一般是输入问题

```
$
time humann2 --input temp/concat/C3.fq  \
>       --output temp/ --threads 8
Output files will be written to: /home/zkzhou/zryuan/meta_ehbio_testdata/temp

Running metaphlan2.py ........

CRITICAL ERROR: Error executing: /home/zkzhou/anaconda2/envs/metagenome_env_ehbio/bin/metaphlan2.py /home/zk
zhou/zryuan/meta_ehbio_testdata/temp/concat/C3.fq -t rel_ab -o /home/zkzhou/zryuan/meta_ehbio_testdata/temp/
C3_humann2_temp/C3_metaphlan_bugs_list.tsv --input_type multifastq --bowtie2out /home/zkzhou/zryuan/meta_ehb
io_testdata/temp/C3_humann2_temp/C3_metaphlan_bowtie2.txt --nproc 8

Error message returned from metaphlan2.py :
Help message for read_fastx.py
(ERR): bowtie2-align died with signal 11 (SEGV) (core dumped)
Error while running bowtie2.
Traceback (most recent call last):
  File "/home/zkzhou/anaconda2/envs/metagenome_env_ehbio/bin/read_fastx.py", line 123, in <module>
    read_and_write_raw(f, opened=False, min_len=min_len)
  File "/home/zkzhou/anaconda2/envs/metagenome_env_ehbio/bin/read_fastx.py", line 89, in read_and_write_raw
    read_and_write_raw_int(inf, min_len=min_len)
  File "/home/zkzhou/anaconda2/envs/metagenome_env_ehbio/bin/read_fastx.py", line 78, in read_and_write_raw_
int
    SeqIO.write(record, sys.stdout, fmt)
  File "/home/zkzhou/anaconda2/envs/metagenome_env_ehbio/lib/python2.7/site-packages/Bio/SeqIO/__init__.py",
 line 557, in write
    fp.write(format_function(record))
IOError: [Errno 32] Broken pipe
```

扫码关注生信宝典，学习更多生信知识

扫码关注宏基因组，获取专业学习资料

易生信，没有难学的生信知识