尊敬的王老师：

展信佳！

本周崭新且丰富的学习生活告一段落，现向您汇报学习收获与进度，请您多多指教。本周只学习了*Bioinformatics Data Skills*这本书，所以就以学习内容作为大序号吧。

一、连接远程服务器

非常凑巧，在您发给我服务器相关资源后，我发现这本书的下一章恰恰谈论远程服务器的连接与技巧，真是无巧不成书。在导入部分，作者亦指出使用远程服务器的意义：生物信息学的分析需要强大的计算能力，有时采用算力强大的服务器才能满足我们的需求。

首先介绍极速登陆的方法。据书中所言，ssh命令是最常用的远程登陆命令，而scp是常用的文件传输命令。要连接服务器，直接使用ssh+账户名@域名（或者IP地址）登陆，默认端口是22，非默认的端口使用-p参数设定。比如我登陆时需要输入：

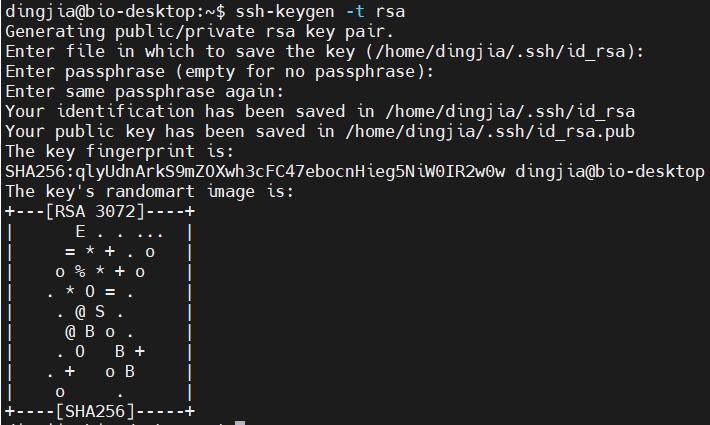
ssh [dingjia@183.169.11.223](mailto:dingjia@183.169.11.223) -p38120

按下回车后会出现输入密码的提示，输入后即可登陆。每次登陆都要输密码固然很麻烦，若要免密登陆则可进行一系列操作实现。

第一步是本地创建身份认证密钥。输入指令：

ssh-keygen -t rsa

而后会弹出多条提示，只需不断按enter就行。接着会随机生成密钥图像，比如：

同时会在/home/way/.ssh目录下生成公开密钥id\_rsa.pub文件与私人密钥id\_rsa文件。公开密钥可放在远程服务器上，而私人密钥需本地妥善保管，泄漏的话会有安全隐患。

第二步是上传公开密钥至远程服务器。在远程服务器创建.ssh文件夹，将公开密钥上传至远程服务器，再设置.ssh目录与authorized\_keys使用权限，便大功告成。

（P.S.在MobaXterm上登陆有另一套流程：打开软件，点击菜单栏中“Tools”选项，在下拉菜单中选择“MobaKeyGen”工具，在弹出的“MobaKeyGen”窗口中点击“Generate”按钮开始生成密钥对。生成后自行保存公钥和私钥至本地。输入如下命令将公钥文件内容追加到授权文件中：

cat'C:\Users\LENOVO\Desktop\dingjia'>>~/.ssh/authorized\_keys

现在，即已配置好了远程主机，可以使用私钥进行免密登录。打开MobaXterm软件，点击菜单栏中的“Session”选项。在下拉菜单中选择“SSH”连接类型。在弹出的“SSH”配置窗口中，输入远程主机的IP地址、用户名和端口号。在“Advanced SSH Settings”选项卡中，勾选“Private key”，并指定私钥文件的路径，点击“OK”按钮建立连接即可成功。）

（P.P.S.其实通过MobaXterm貌似正常登陆一次后便可直接快捷免密登陆。依次点击Session,SSH选项，输入Hostname和Username再调整Port至38120，输入密码后便可登陆。此后再进入只需点击服务器名称即可。）

二、维持程序长期运行

在服务器上长时间运行程序的情况应当很常见，问题在于即使后台运行程序，但是当关闭终端或者网络波动时都会触发挂起SIGHUP信号并中断程序，非常不人性化。作者介绍了两种方式来避免这种问题，分别是nohup命令与screen工具。

1.nohup命令(no hang up)

nohup命令的使用方式非常直接,例如：

nohup 文件名 &

这样就能成功实现程序持续运行。注意成功使用nohup运行程序后不要直接关闭终端，而是采用exit命令退出，否则程序还是会断。nohup成功运行后下面显示的数字为PID，退出并重新回到终端时jobs命令已经失效，终止程序只能使用kill PID命令。

2.screen工具

这似乎是一个更加“柔和”的工具，类似的还有tmux。使用如下命令查看版本：

screen --version

基础操作并不复杂。命令行输入screen（会弹出用户协议，按enter跳过），然后输入想要执行的程序即可。如果想要终止一个程序，使用screen -r回到该程序，使用快捷键ctrl+a k，输入y确认即可。

三、GIT初识

GIT是一个开源的分布式版本控制系统，可以有效、高速地处理从很小到非常大的项目版本管理，是一款为了帮助管理Linux内核而开发的开放源码的版本控制软件。GIT有以下作用：

1.提供工程的镜像

生成工程的镜像可以避免犯错造成的损失，可以随时进行版本回滚。同时，生成镜像可以更加容易地寻找BUG，例如一个结果第二次运行的时候结果和第一次不同，那么根据历史记录能知道结果是在何时改变的。这也能提供更好的记录方式，例如若你对结果进行了改动与覆盖，亦可以完整地重现。

2.提供重要改动的追踪。

可以及时地追踪代码中使用的软件最新的版本。

3.使工程更加有组织性

一个GIT库作为一个工程将所有的东西放在一起，会更有组织性。另外，如果工程需要分享给别人的话也更加便捷，利于工作交接。

四、GIT基操

1.“告诉它我是谁”

对git进行身份配置的命令如下：

git config --global user.name"dingjia"

git config --global user.email[3151710452@qq.com](mailto:3151710452@qq.com)

方便区分可设命令颜色。例如，红色表示删除，绿色表示新的或修改的内容。命令如下：

git config --global color.ui true

2.git初始化

git初始化有两种方式。第一种是基于已经存在的文件夹，命令很简单（注意前提是以管理员身份打开git）：

git init

第二种方式是直接复制别人的库，作者的例子如下：

git clone https://github.com/lh3/seqtk.git

3. git文件track和stage

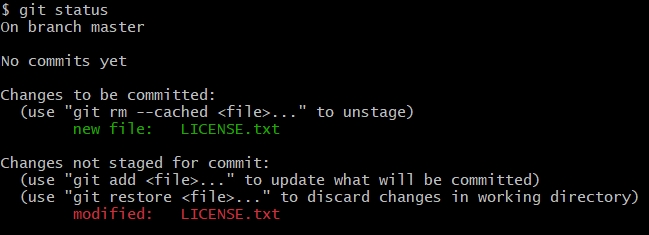
git 不会自动开始跟踪目录中的每个文件。相反，需使用子命令 git add告诉git要跟踪哪些文件。这实际上是git的一个实用功能——生物信息学项目包含许多不必跟踪的文件，包括大数据文件、中间结果或任何可以通过重新运行命令轻松重新生成的文件。

要追踪文件，首先使用git status查看状态，命令如下：

git status

此时出现的信息nothing added to commit but untracked files present说明没有文件是被追踪的，可以通过git add命令来添加追踪文件。手动添加虽然麻烦，但是可以避免追踪一些不必要文件,例如大的原始数据、中间数据。以LICENSE.txt这一文件为例：

git add LICENSE.txt

这样即可追踪该文件，再次使用git status可查看状态。同时，git add命令也可暂存对已跟踪文件所做的更改（暂存的更改将包含在下一次提交中）。第一次git add让git知道一个文件的存在并生成第一次存档，后面对文件进行修改但是没有改完的话可以不进行更新，等到将这个文件修改完毕了再使用git add把这个文件加入更新列表，下次提交的时候才生成快照。

使用git commit命令可生成工程快照，包含写给协作者（以及将来的自己）关于特定提交内容的注释。命令如下：

git commit -m "信息"

其中-m可以不加。如果不加-m会打开默认编辑器。

五、一点问题

在登陆服务器后，我确实可以使用lunix命令行来模拟书中指令。但是总感觉现在学的知识有些虚无缥缈，不知道具体能用来干嘛，说不定过段时间还会遗忘已经所学的。可能是没能和实际的生信程序联系起来，导致印象不太深刻，也不知道实际工作中是怎么运用知识的。其实，我甚至还不知道生信程序通常是用来进行什么工作的。这种脚不踏实地的感觉让我有点犹豫和担心，不知道老师怎么看。

六、规划

Python确定是先不学了（笑）。

*Bioinformatics Data Skills*这本书我会尽我所能地抽时间推进，下周继续学习GIT。可喜的是现在进度已经达到1/5了（好像也不是很多）。

感谢老师每周耐心看我的学习报告，我会继续努力，还请多多指教，有错误的话也请多多斧正，谢谢！

丁加

2024.12.8