



北京大学前沿计算研究中心
Center on Frontiers of Computing Studies, Peking University

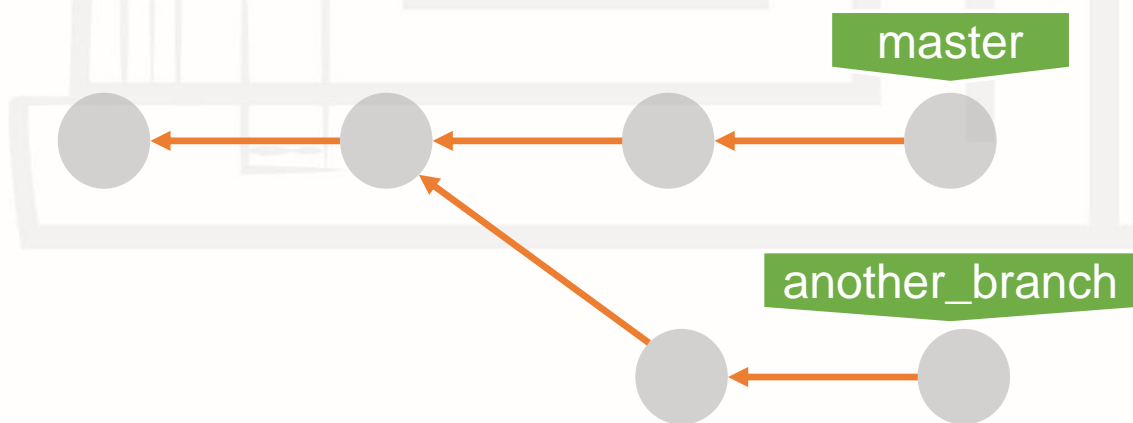
科研工具介绍

姚贺源

导师：刘利斌 陈宝权

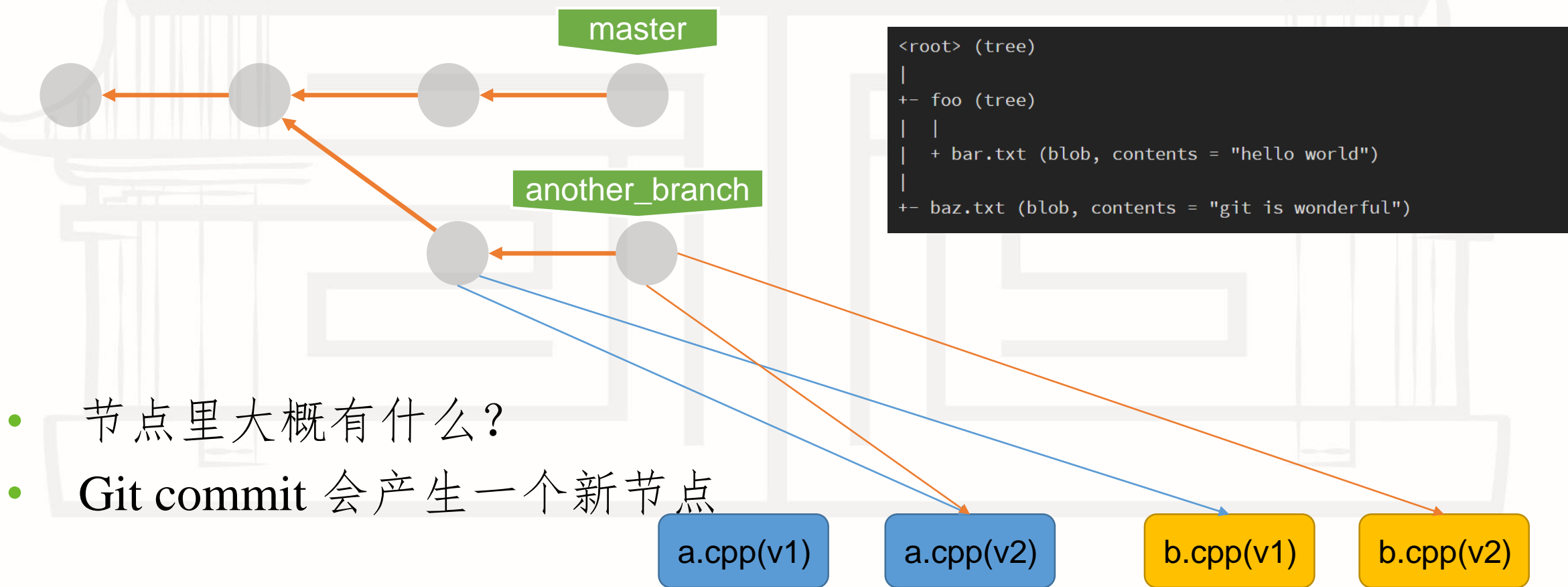
Git

- **Git:** 版本控制软件
 - 对不同时间的文件夹情况拍一个快照
 - 用时间和逻辑关系组织快照，附上修改者和修改时间等信息
 - 同一个**Git**仓库可以保存在不同的地方，互相之间可以毫无干系



- 节点：保存的版本快照
- 指针：一个分支，指针可以移动

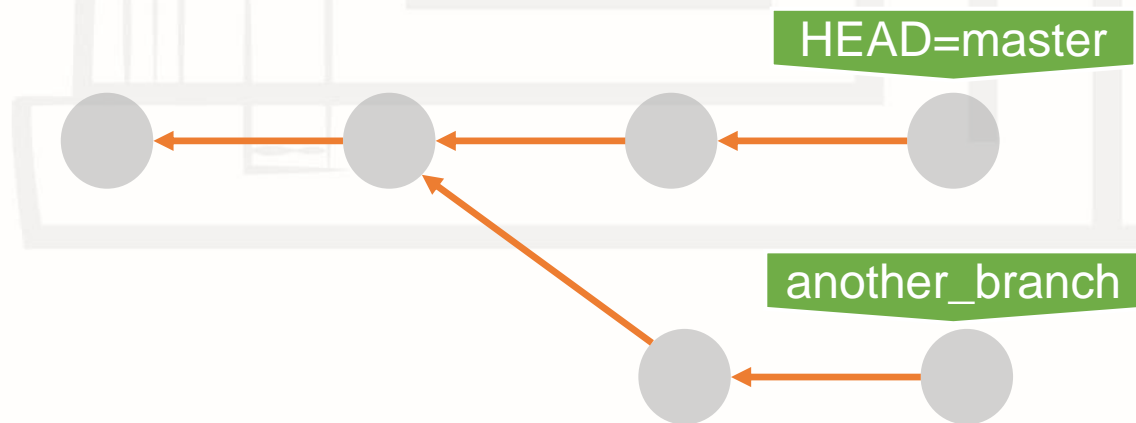
Git commit



- 节点里大概有什么？
- Git commit 会产生一个新节点

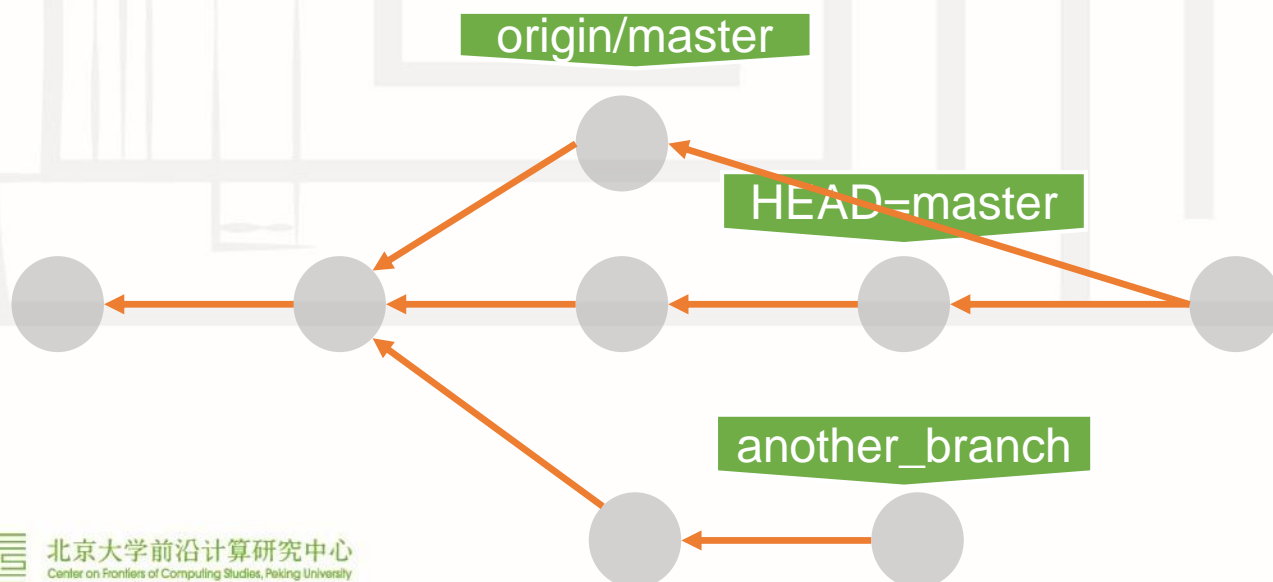
Git branch and checkout

- Git: 版本控制软件
 - Master等指针用来标志着每个分支的末尾
 - Head指针表示当前处于哪个节点（可以指向指针或节点）
 - Checkout: 移动Head指针



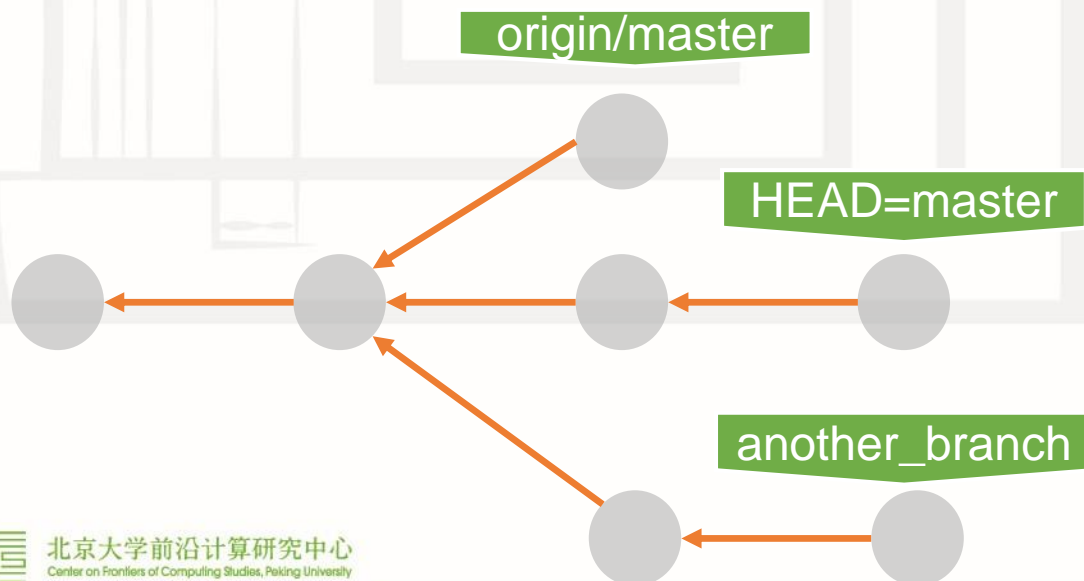
Git merge

- Git merge 含义是在当前分支创建一个新节点，让其包含另一个分支的更改
- 如果当前分支是另一个分支的子集，会Fast Forward



Git缩并命令

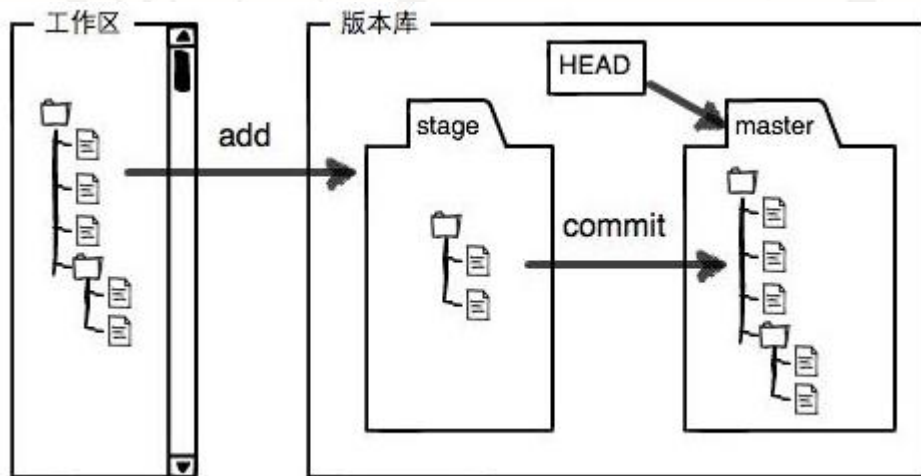
- `git checkout -b` = `git branch` + `git checkout`
- `git pull` = `git fetch` + `git merge` (fetch: 获取远程分支)



- `merge`: 创建新节点, 合并更改

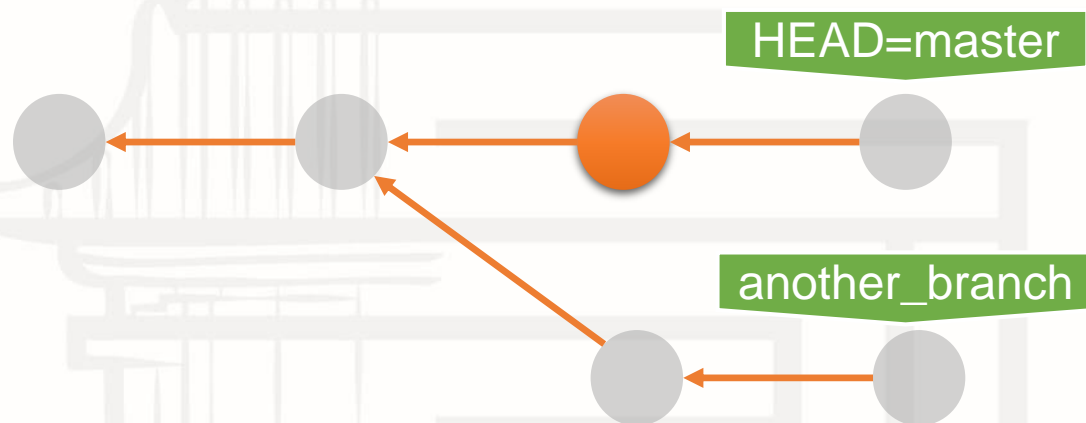
Git暂存区

- 更改后不会立刻生成副本（否则代价太大了）
- 不是所有更改都应该被同步（比如一些私人配置和Debug代码）
- 需要手工指定哪些更改要被保存（staged）



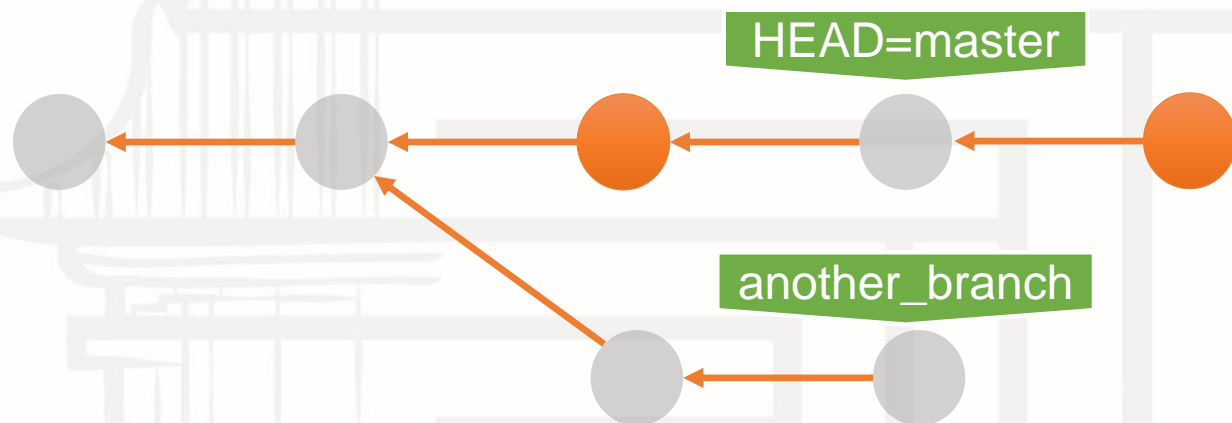
- **add:** 将更改保存进暂存区
- **commit:** 将暂存区的更改提交
- **git commit -a:** 暂存所有更改并提交

Git Reset & Revert



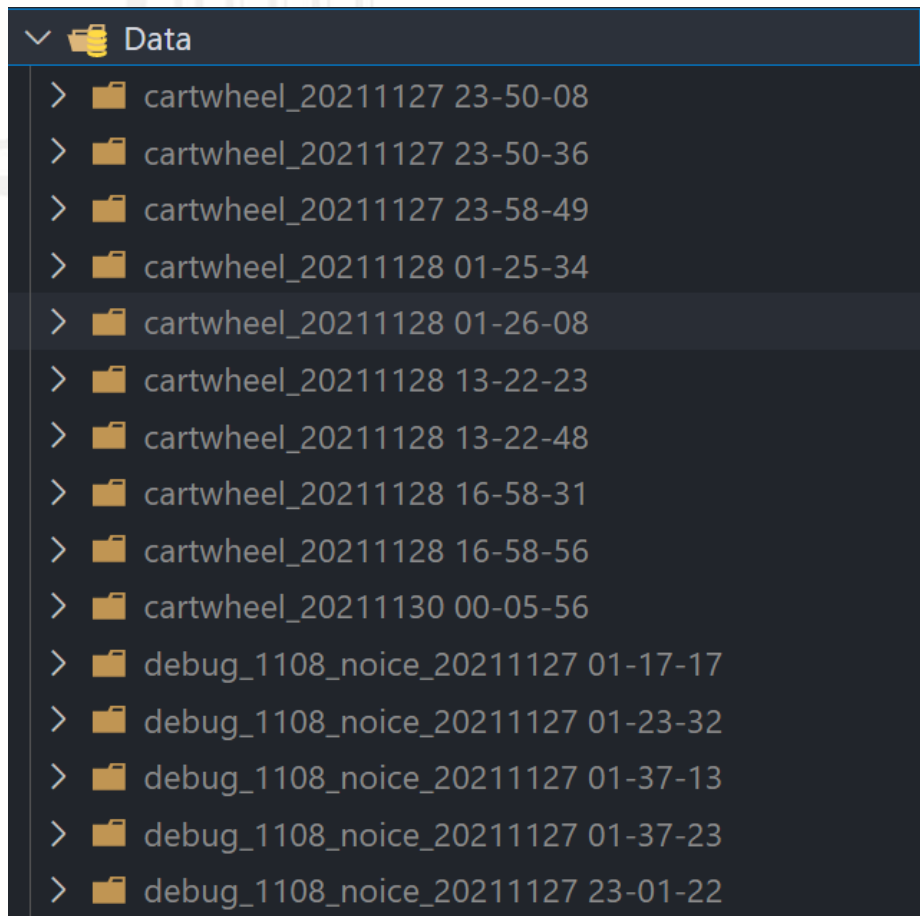
- Git Reset: 将最后一次的更改存入缓存区，然后指针回退一格
- Git Revert: 将撤销更改这次更改提交为一个新的节点

Git Reset & Revert

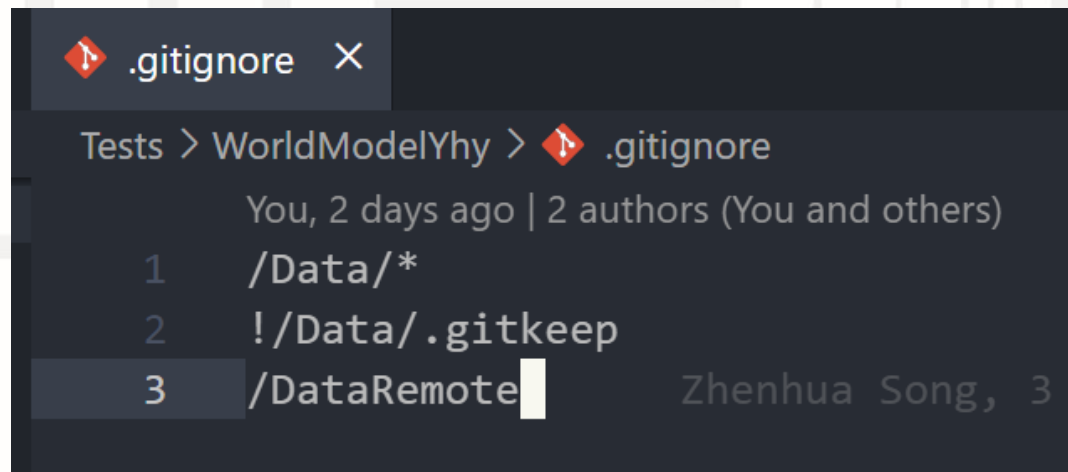


- Git Reset: 将最后一次的更改存入缓存区，然后指针回退一格
- Git Revert: 将撤销更改这次更改提交为一个新的节点

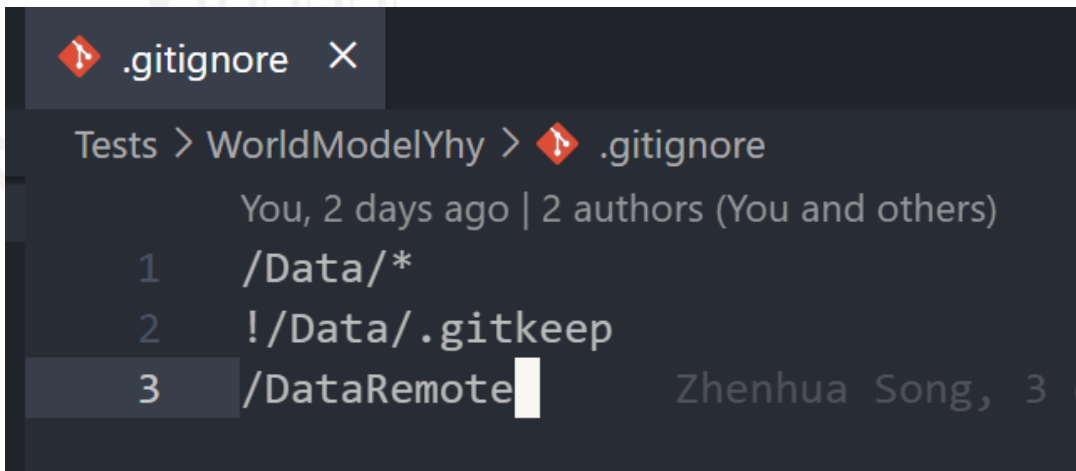
Git keep & Git ignore



- Data文件夹下的数据都是程序生成的，理论上不应该放入git
- Git ignore: 记录Git可以忽略哪些文件的更改



Git keep & Git ignore

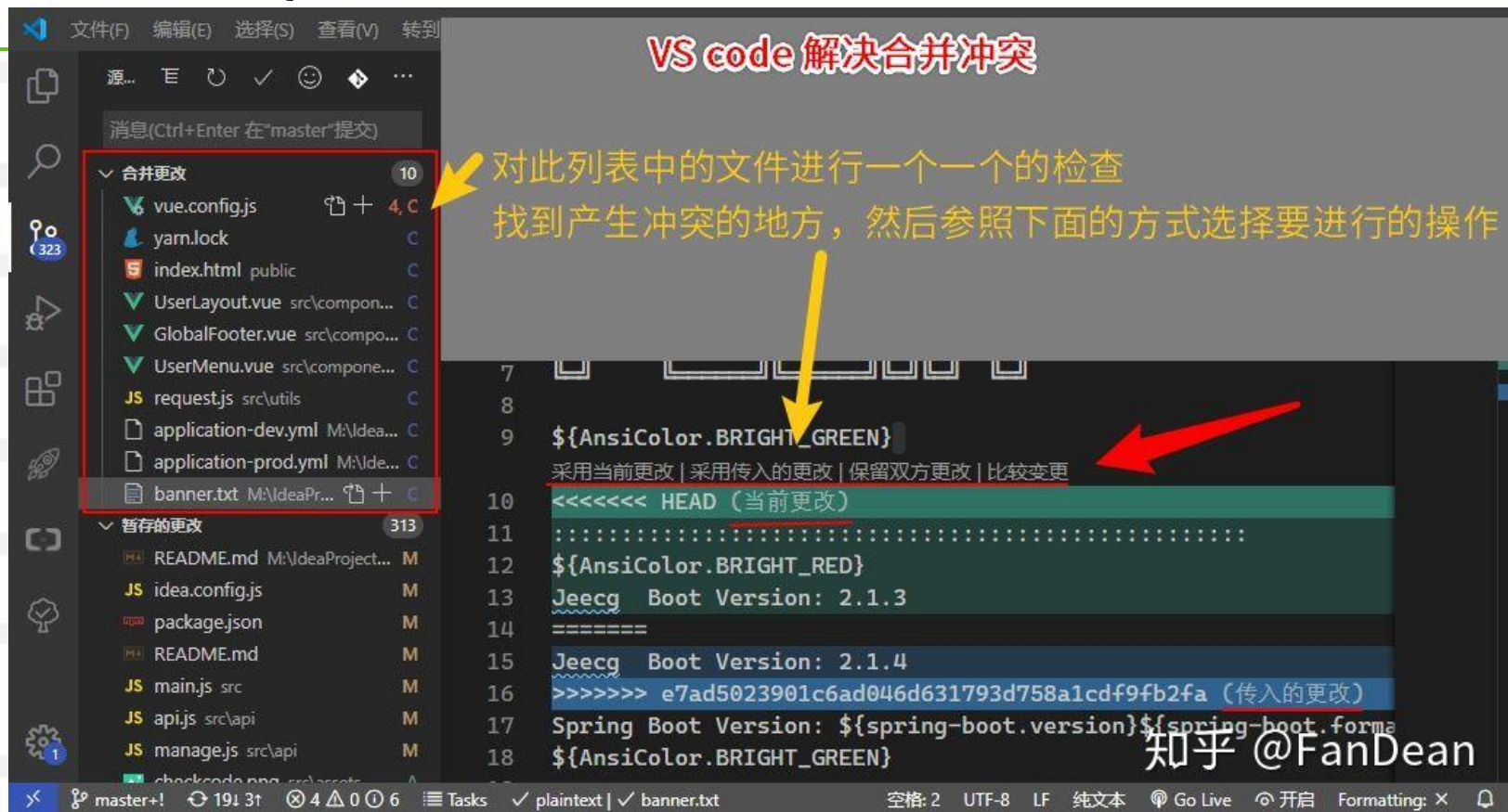


```
.gitignore
Tests > WorldModelYhy > .gitignore
You, 2 days ago | 2 authors (You and others)
1 /Data/*
2 !/Data/.gitkeep
3 /DataRemote
```

- 如果只用第一行，会发现不仅是Data下的文件被忽略了，Data文件夹本身也从Git消失了
- 这是因为Git追踪以文件为基础。一般为了保留空文件夹会在该文件夹下创建一个.gitkeep文件
- Ignore中的!表示不ignore这个

Git UI(处理conflict)

- 直接找的图片



- <https://zhuanlan.zhihu.com/p/126137780>



更多Git

- 常用指令如 `git status`, `git log`, `git diff`, `git mv`, `git blame`
- Conflict 的处理, 分支管理
- 即便有Git, 多人合作依然很艰难, 使用时需要谨慎, 防止误删他人代码或引入bug
- https://learngitbranching.js.org/?locale=zh_CN

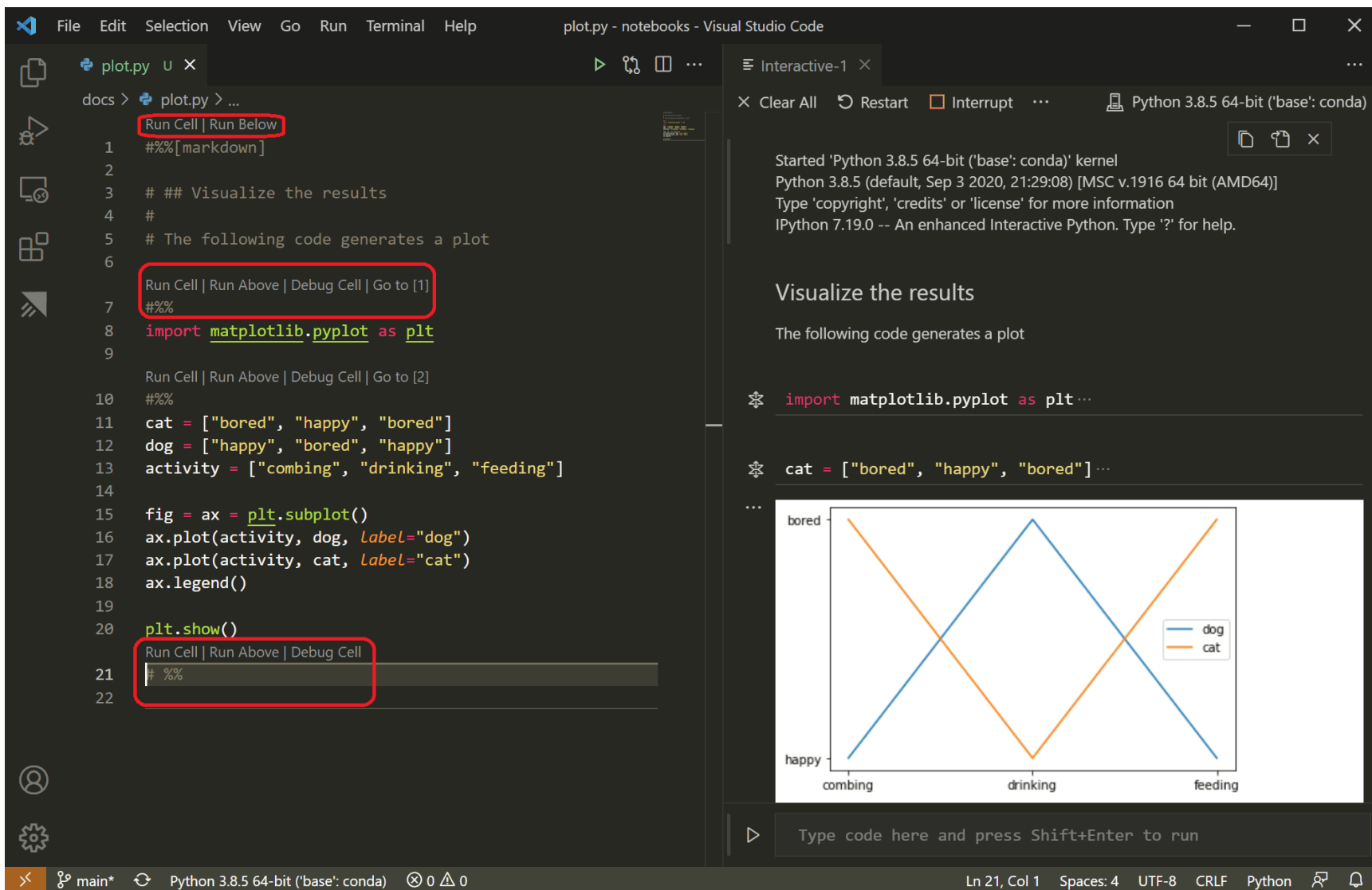


Some Python Trick: 使用Anaconda

- 经常使用他人代码/开发不同环境
- 安装的Python版本不同，包不同
- Anaconda提供虚拟环境，每个环境内可以有独立的Python 版本和包管理
- 下载和安装可以自查，唯一建议使用国内镜像站（如清华源）



Some Python Trick: 尝试jupyter notebook



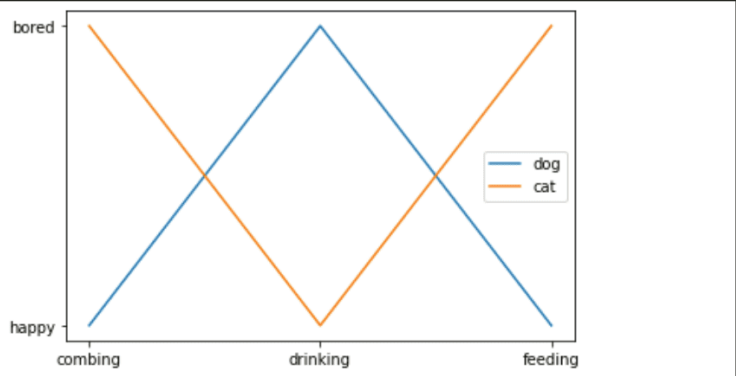
The screenshot shows the Visual Studio Code interface with a Jupyter Notebook open. The notebook has three cells. The first cell contains a markdown header. The second cell contains a markdown paragraph. The third cell contains Python code that imports matplotlib.pyplot as plt, defines two lists (cat and dog) representing animal states, and plots these states against activities (combing, drinking, feeding). The plot shows two lines: a blue line for 'dog' and an orange line for 'cat'. The 'dog' line starts at 'happy' for 'combing', goes to 'bored' for 'drinking', and returns to 'happy' for 'feeding'. The 'cat' line starts at 'bored' for 'combing', goes to 'happy' for 'drinking', and returns to 'bored' for 'feeding'. The plot is titled 'Visualize the results' and includes a legend.

```
1  #%%[markdown]
2
3  # ## Visualize the results
4  #
5  # The following code generates a plot
6
7  #%%
8  import matplotlib.pyplot as plt
9
10 #%%
11 cat = ["bored", "happy", "bored"]
12 dog = ["happy", "bored", "happy"]
13 activity = ["combing", "drinking", "feeding"]
14
15 fig = ax = plt.subplot()
16 ax.plot(activity, dog, Label="dog")
17 ax.plot(activity, cat, Label="cat")
18 ax.legend()
19
20 plt.show()
21 #%%
22
```

Visualize the results

The following code generates a plot

```
* import matplotlib.pyplot as plt ...
* cat = ["bored", "happy", "bored"] ...
```



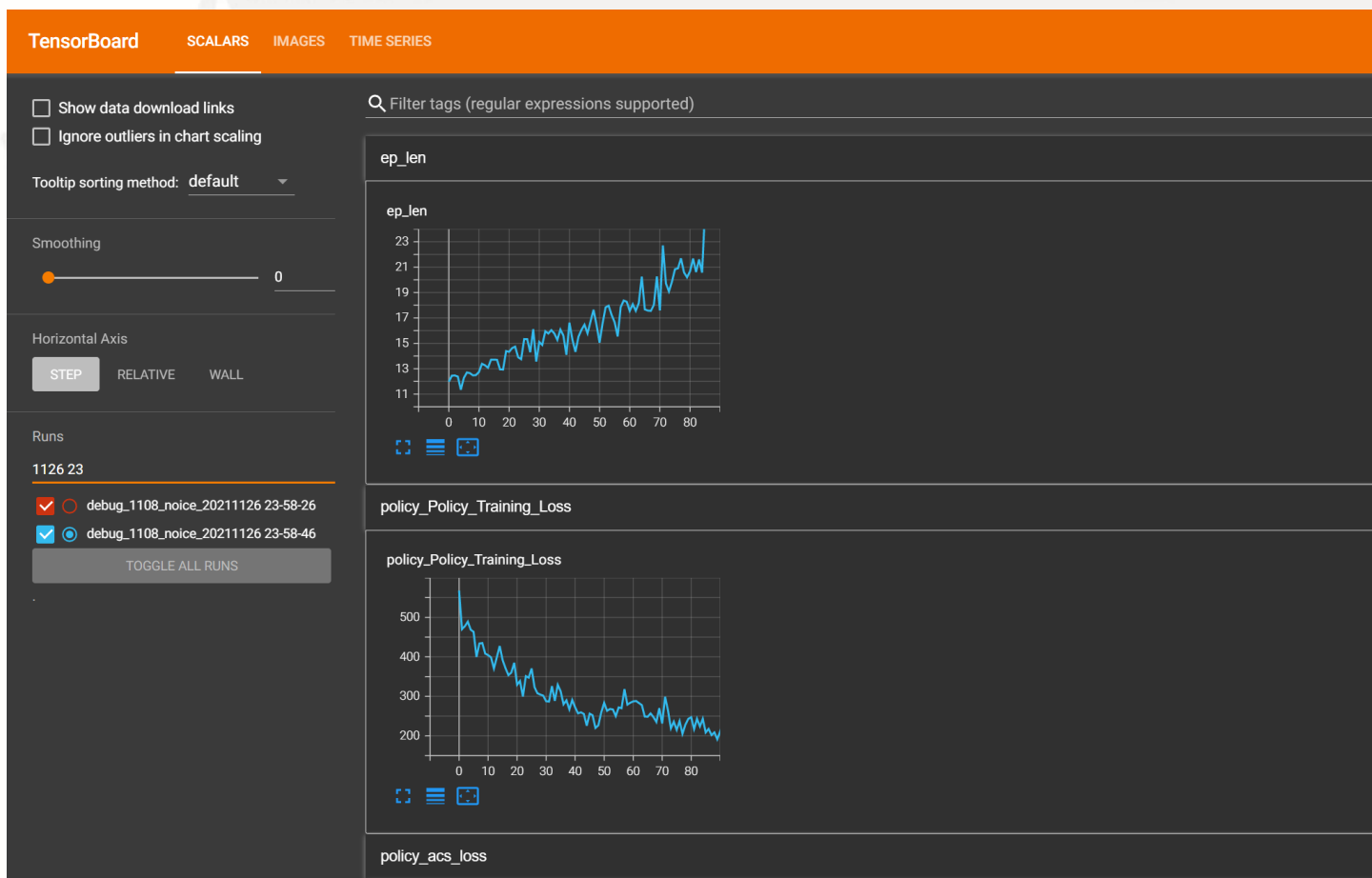
combing drinking feeding

dog cat

Type code here and press Shift+Enter to run

支持
python 代码块,
markdown 文本块,
Matplotlib 等可视化

Some Python Trick: 使用Tensor board



每个epoch都记录到Tensorboard的event文件中(用logger)

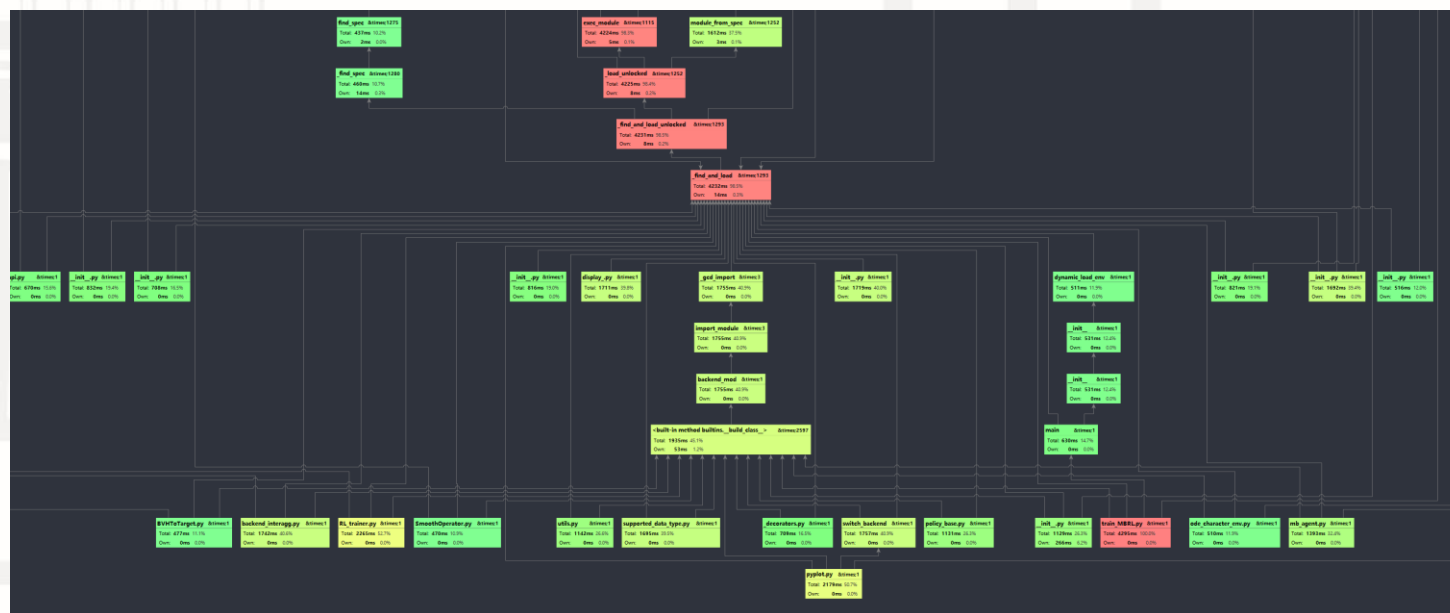
在命令行里输入
`tensorboard -logdir xxx`(event文件的父目录)

在浏览器里打开上一步返回的端口

可以记录变量, 图片

Some Python Trick: 使用Profile

- 代码写完了以后需要检查错误和估计效率



Some Python Trick: 使用Profile

- 代码写完了以后需要检查错误和估计效率

<built-in method _imp.create_dynamic>	125	1538	35.8%
<built-in method io.open_code>	1121	391	9.1%
<built-in method nt.stat>	5119	278	6.5%
init.py	1	1129	26.3%
<built-in method marshal.loads>	1109	229	5.3%
_path_join	9852	141	3.3%
<method 'read' of '_io.BufferedReader' objects>	1115	58	1.4%
cleandoc	2701	87	2.0%
<built-in method builtins._build_class_>	2597	1935	45.1%
docformat	339	101	2.4%
<built-in method _imp.exec_dynamic>	125	711	16.6%
find_spec	2024	404	9.4%
get_data	1121	479	11.2%
<method 'sub' of 're.Pattern' objects>	692	27	0.6%
<method 'startswith' of 'str' objects>	66143	22	0.5%
get_code	1115	839	19.5%
<built-in method builtins.len>	135797	19	0.4%
<method 'findall' of 're.Pattern' objects>	358	18	0.4%



Cmake(Building System)

- **Make**, Cmake, cmake
- **Make:**
- 类似于脚本，每次调用它都是为了“生成”一些目标(exe, gif, pdf)
- 把长长的生成命令如
 - `gcc -balabla xxxx ...` 写进去，可以一键生成
- 但是还会做更多
 - 比如生成一个pdf之前需要先生成一个图片
 - 比如图片之前生成过了，判断是否需要重新生成



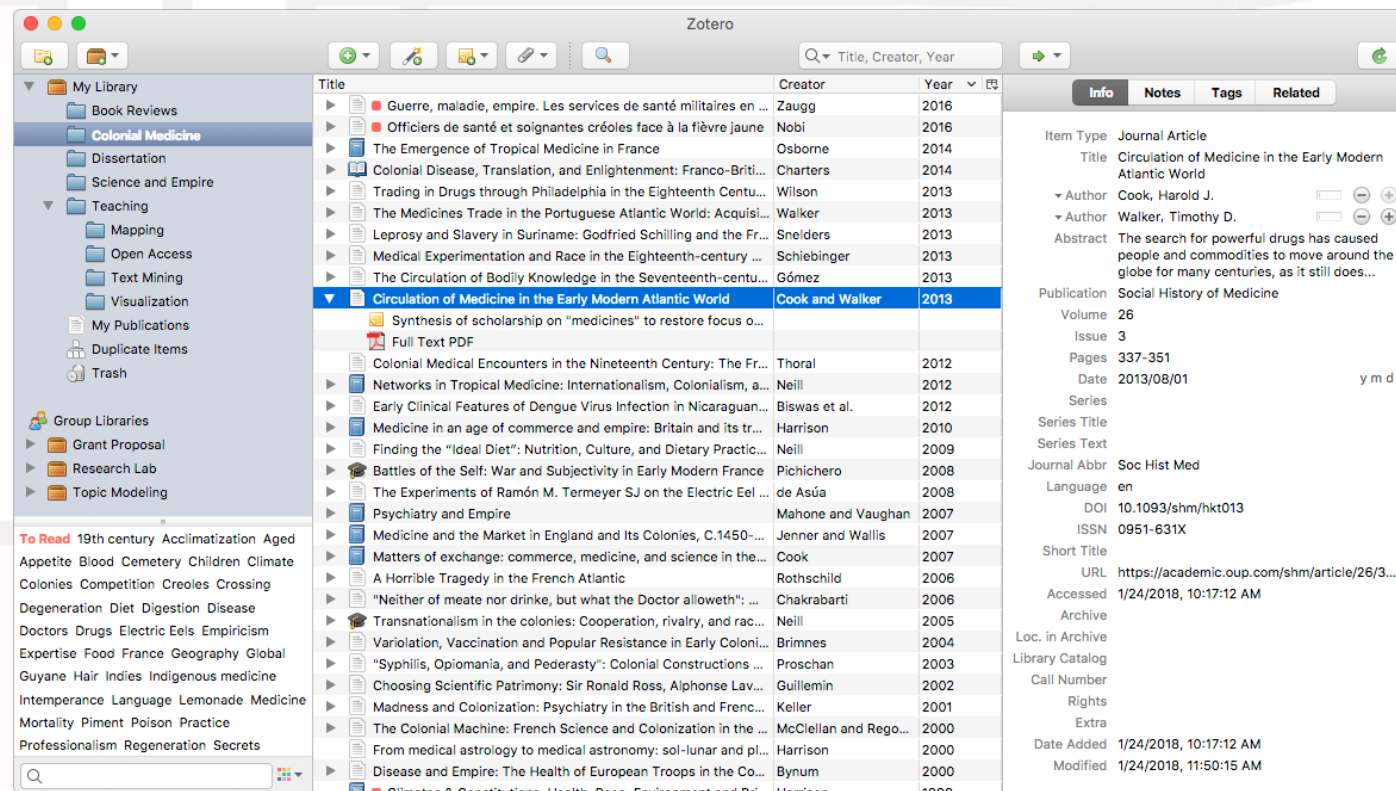
Cmake(Building System)

- **Make**, **Cmake**, **ccmake**
- Cmake:
- Make的生成命令需要自己手敲进去，并且依赖于环境
 - `gcc -c xxx.cpp -o xxx`
- Cmake帮你生成这条命令，从而你可以用更抽象的语言
 - `add_executable(xxx xxx.cpp)`
- 好处是简洁且跨平台



Zotero 文献管理

- 可以收录文章的pdf, meta data, 可以添加笔记
- 类似软件还有Mendeley, Endnote
- 方便导出引用(bibtex...)





北京大学前沿计算研究中心
Center on Frontiers of Computing Studies, Peking University

感谢倾听



Q&A