Lab1: 简单集群搭建

1. 概况

实验环境：`macOS 26.5.1, Debian GNU Linux`

使用到的Ask/Agent：`GitHub Copilot: ChatGPT-5.x & deepseek v4-pro`

通过此次环境配置和小集群的搭建，我意识到了AI的日益强大，也意识到自己亟需掌握更多Linux环境的知识和经验。文档的信息量不小，我需要慢慢消化。

> 另外，这个文档只记录了最基础的部分，将在期末考试后有所更新。
>
> 应该有挺多感想和挫折的，但是现阶段时间紧张，留到考试之后写。
>
> 所以文档中目前主要是记录配置过程。

2. 任务一：从源码构建 OpenMPI、BLAS 和 HPL

2.1 OpenMPI的构建

- 下载并`scp`压缩包

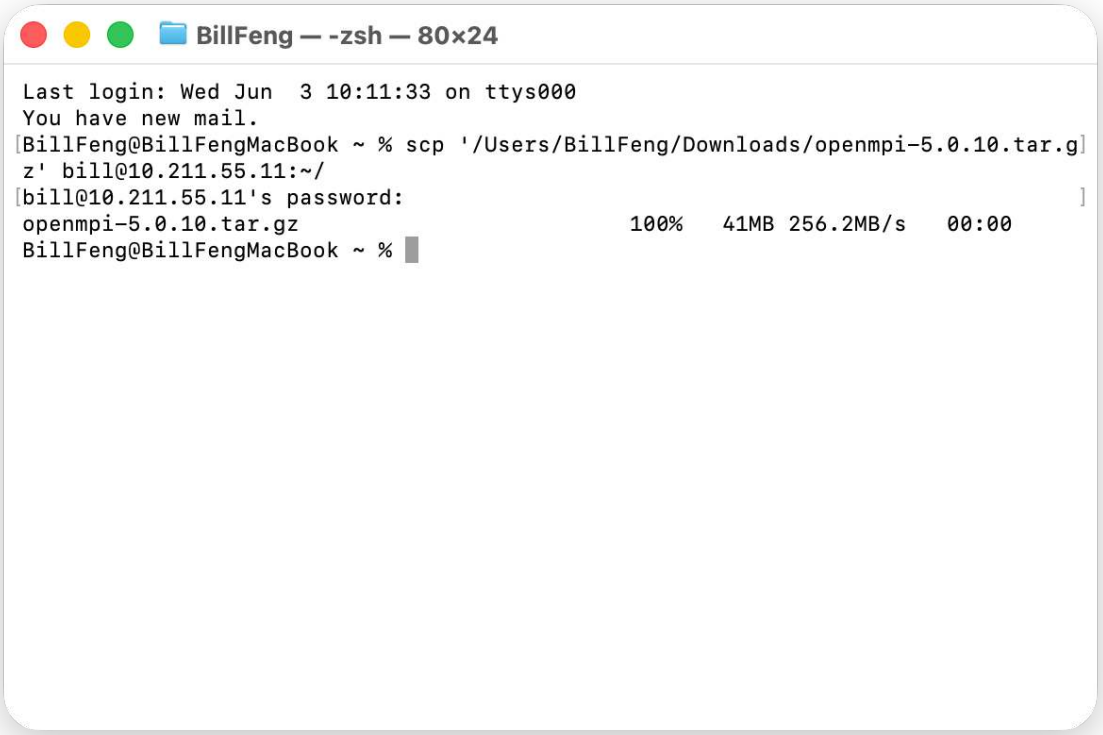

Image 1

- 解压并安装

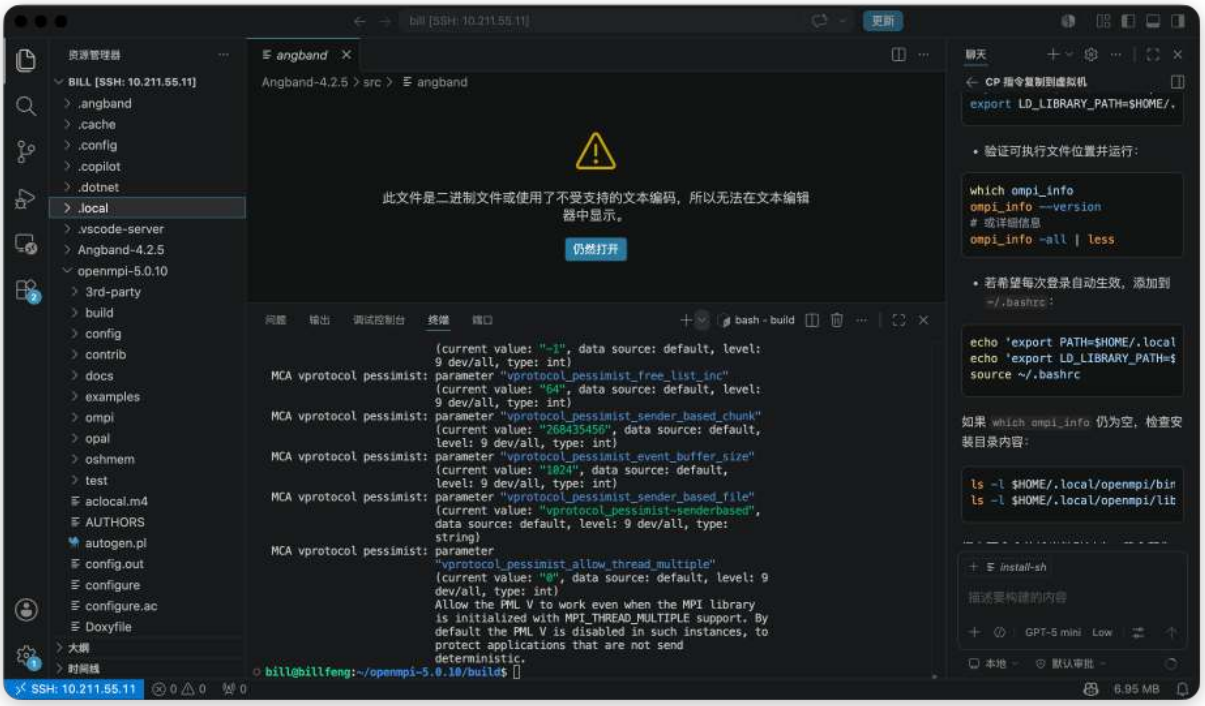

Image 2

2.2 BLAS&CBLAS

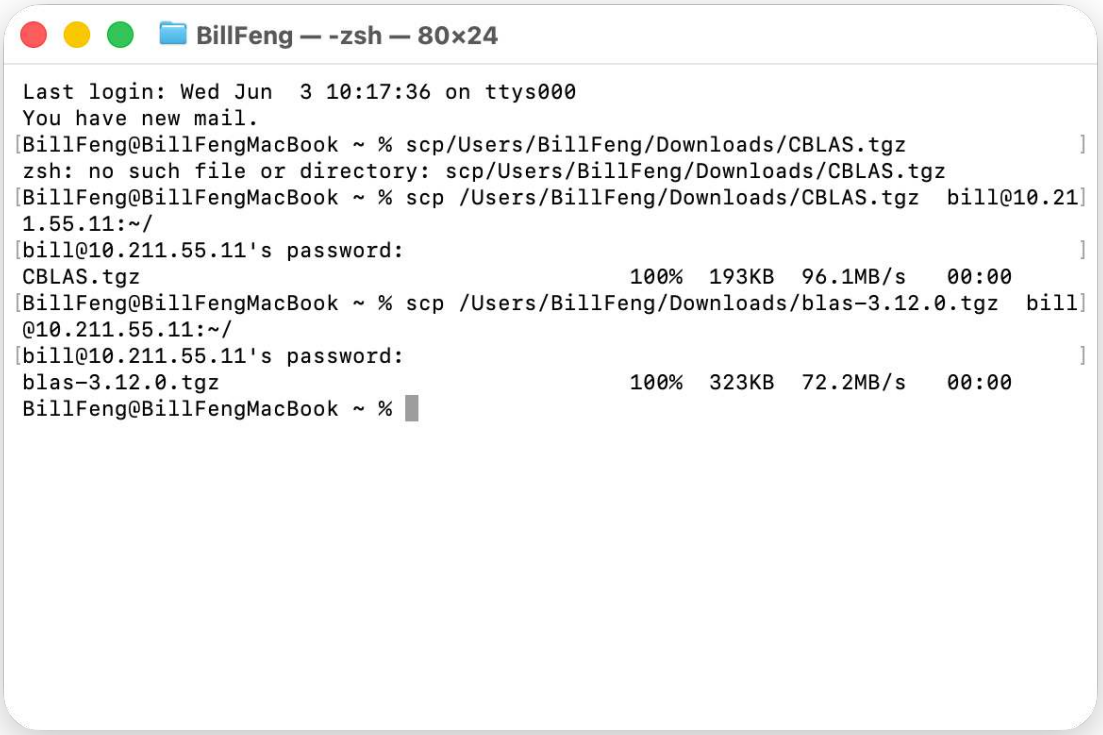

Image 3

- 修改编译器配置:

```
bill@billfeng:~$ cd BLAS-3.12.0
bill@billfeng:~/BLAS-3.12.0$ make
gfortran -O2 -frecursive  -c -o isamax.o isamax.f
make: gfortran: No such file or directory
make: *** [<builtin>: isamax.o] Error 127
```

Image 4

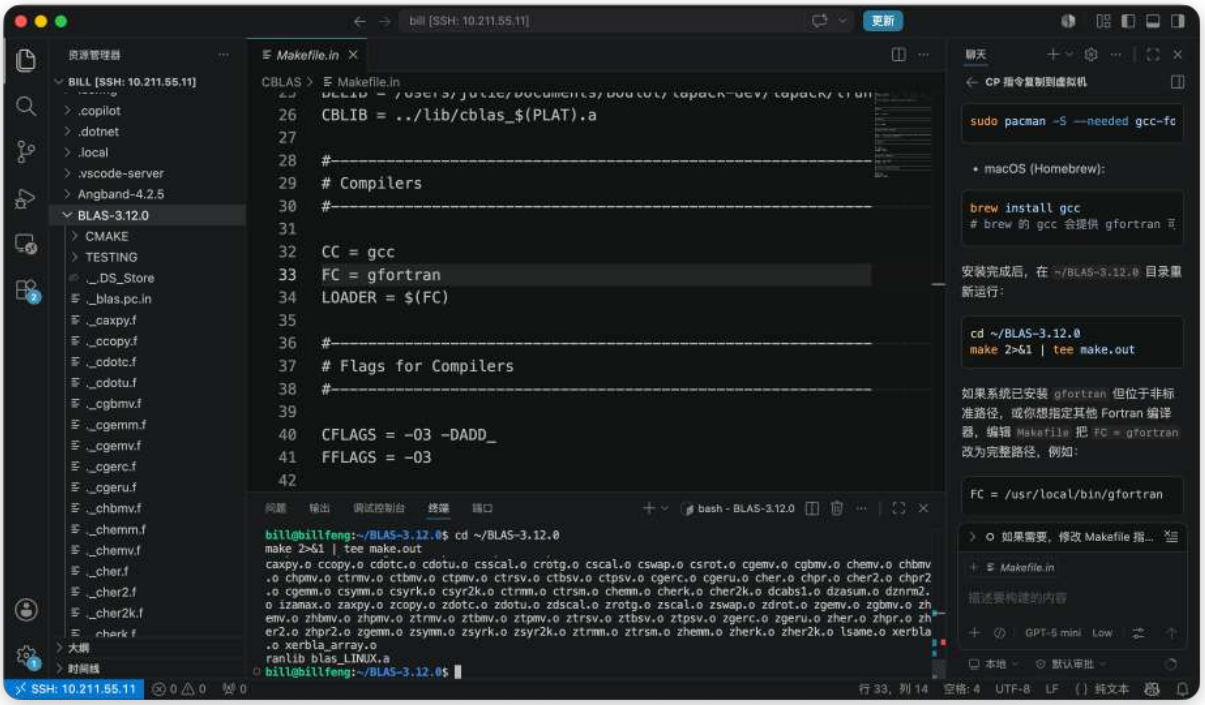

Image 5

- 遇到了文档中的问题，修改 `TOPdir` 路径：

```
bill@billfeng:~$ wget "https://netlib.org/benchmark/hpl/hpl-2.3.tar.gz"
tar xvf hpl-2.3.tar.gz
cd hpl-2.3
cp setup/Make.Linux_PII_FBLAS . …
make[1]: Entering directory '/home/bill/hpl-2.3'
( cd src/auxil/Linux_PII_FBLAS;         make )
make[2]: Entering directory '/home/bill/hpl-2.3/src/auxil/Linux_PII_FBLAS'
Makefile:47: Make.inc: No such file or directory
make[2]: *** No rule to make target 'Make.inc'.  Stop.
make[2]: Leaving directory '/home/bill/hpl-2.3/src/auxil/Linux_PII_FBLAS'
make[1]: *** [Make.top:54: build_src] Error 2
```

Image 6

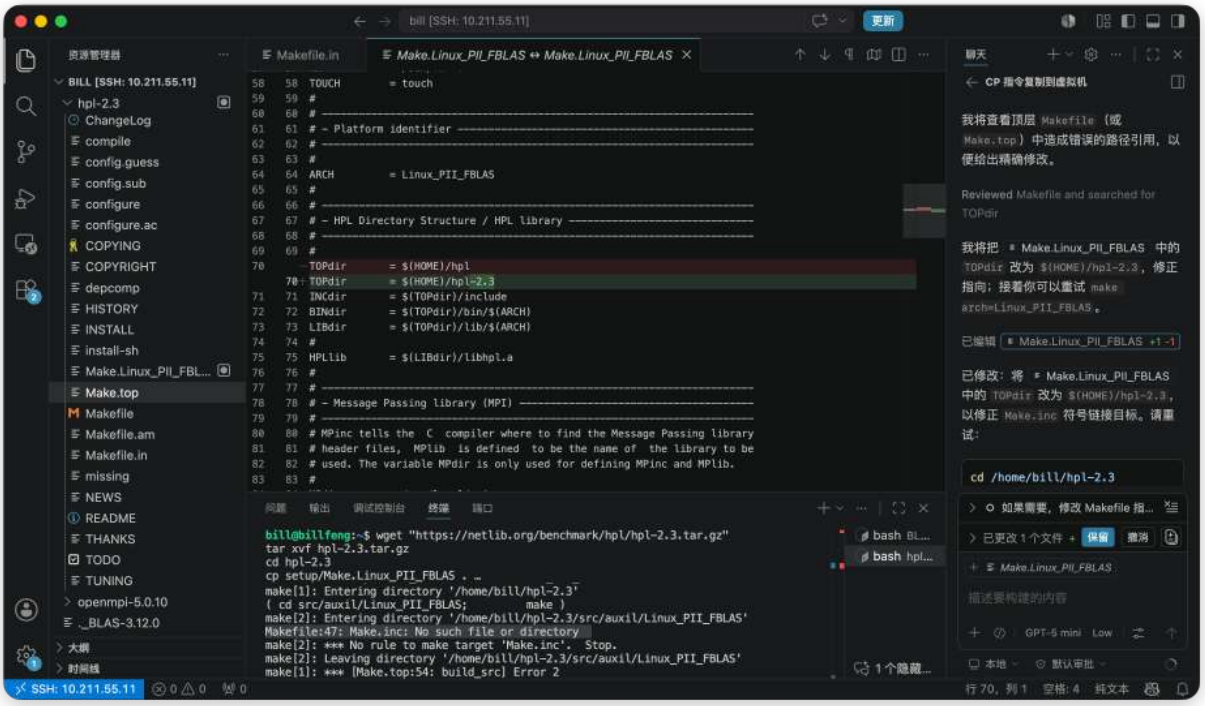

Image 7

2.3 HPL

下面是在 `Make.Linux_PII_FBLAS` 中修改或设定的参数：

- `TOPdir` : `$(HOME)/hpl-2.3`
- `MPdir` : `/home/bill/.local/openmpi`
- `MPinc` : `-I$(MPdir)/include`
- `MPlib` : `$(MPdir)/lib/libmpi.so`
- `LAdir` : `$(HOME)/BLAS-3.12.0`
- `LAlib` : `$(LAdir)/blas_LINUX.a $(LAdir)/cblas_LINUX.a`
- `LINKER` : `/usr/bin/gfortran`

成功生成可执行文件在 `bin/Linux_PII_FBLAS` 目录下：

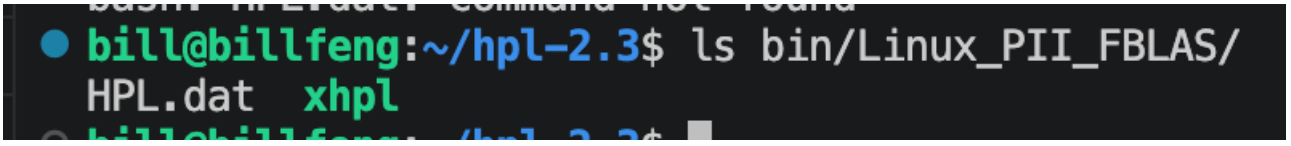

Image 8

3. 任务二：配置 NFS 共享目录

- 这是四个节点映射的IP地址：

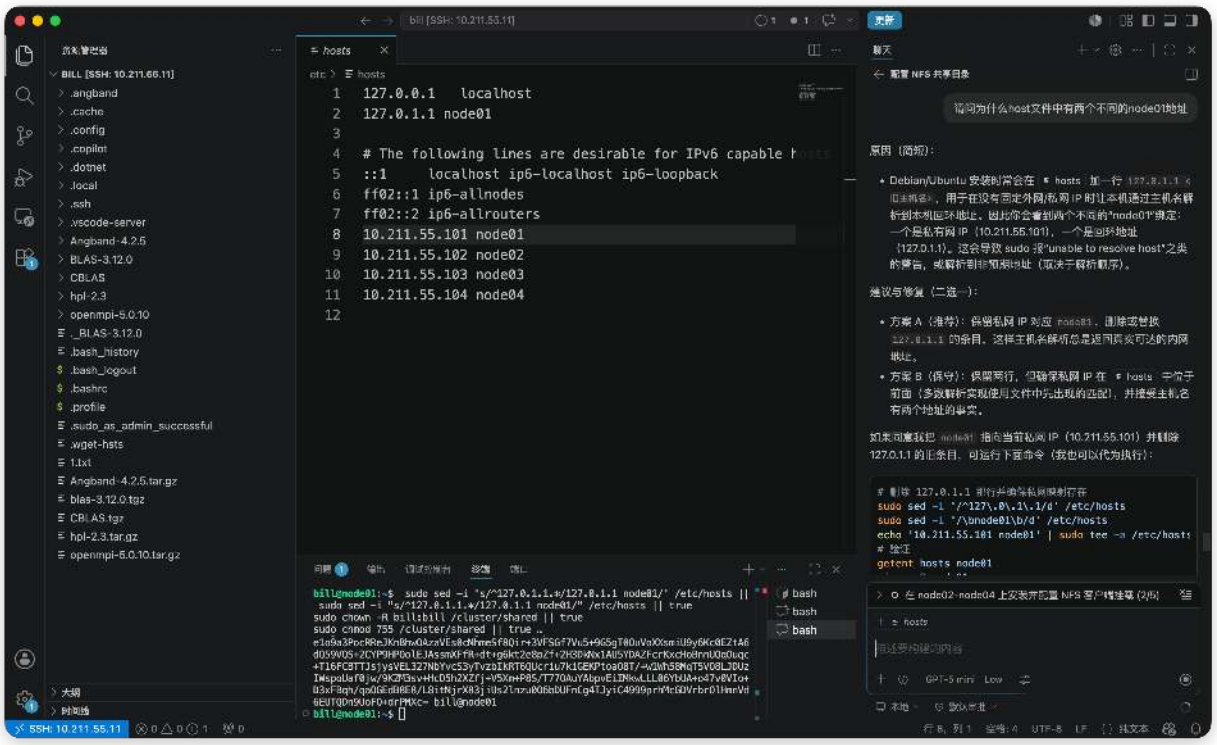

Image 9

- 我的思路是先配置 `node02`，然后将配置好的节点拷贝两份。下面三张图是我在 `node02` 中操作的记录。

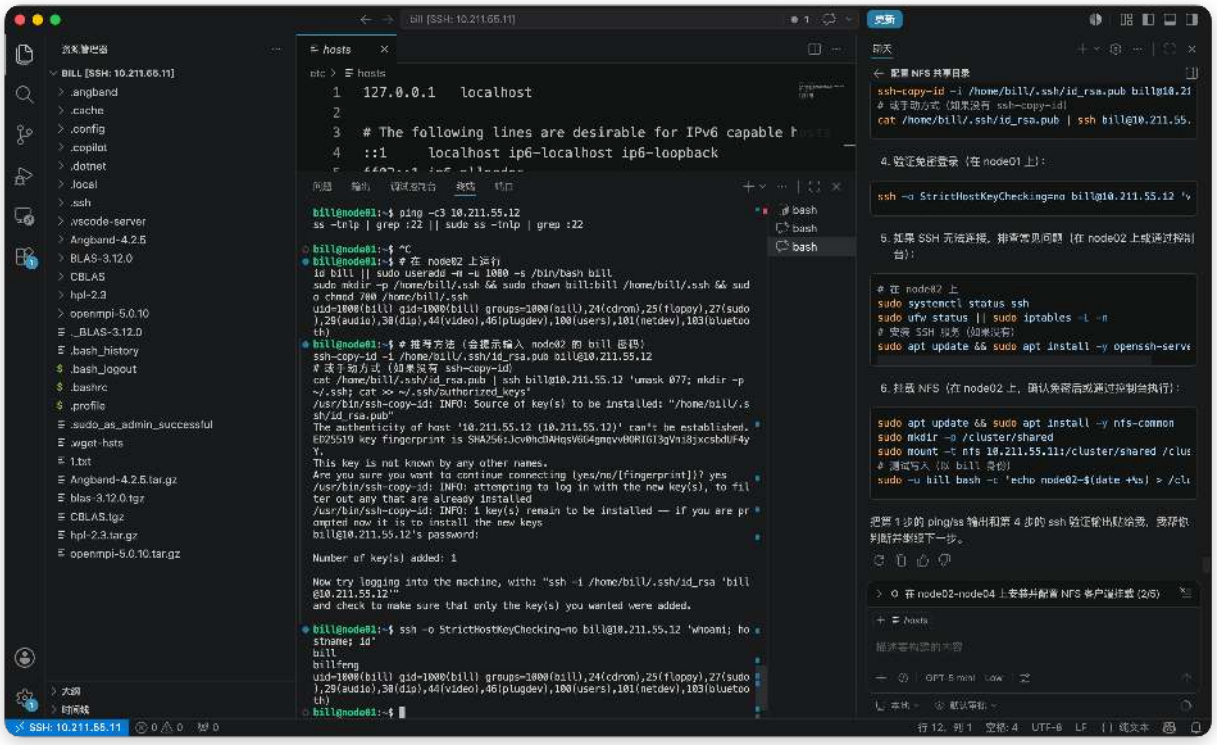

Image 10

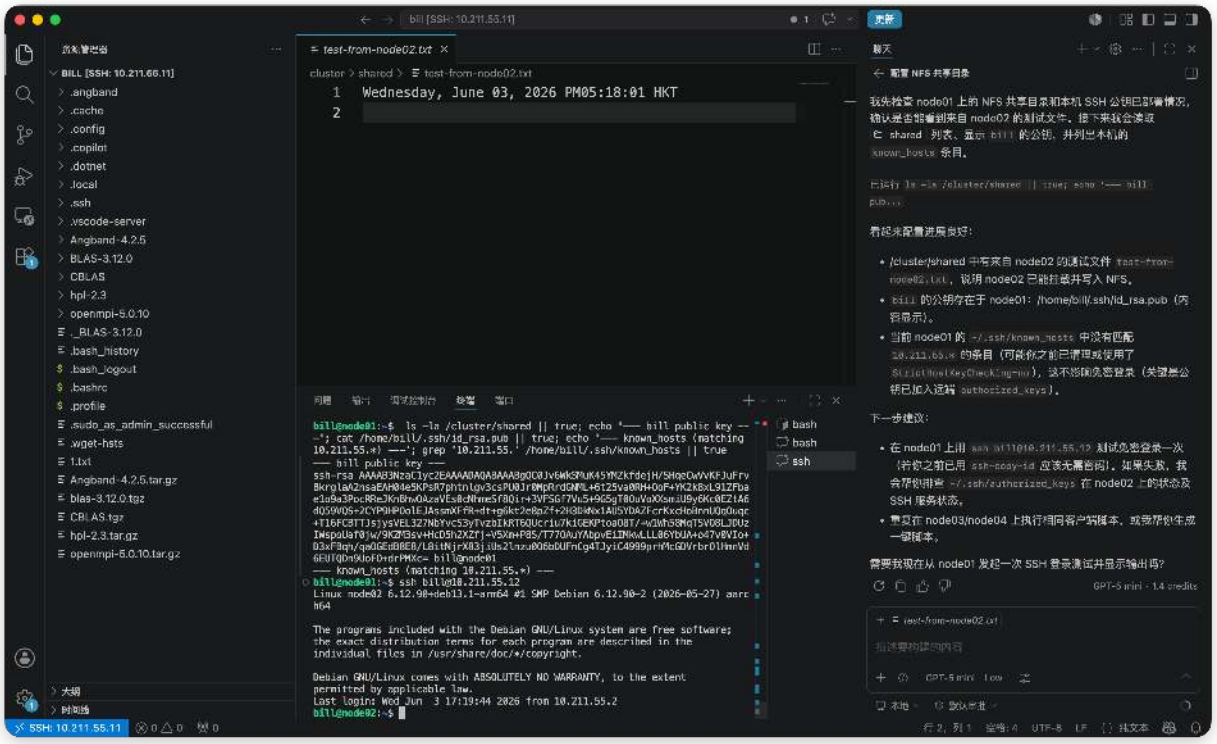

Image 11

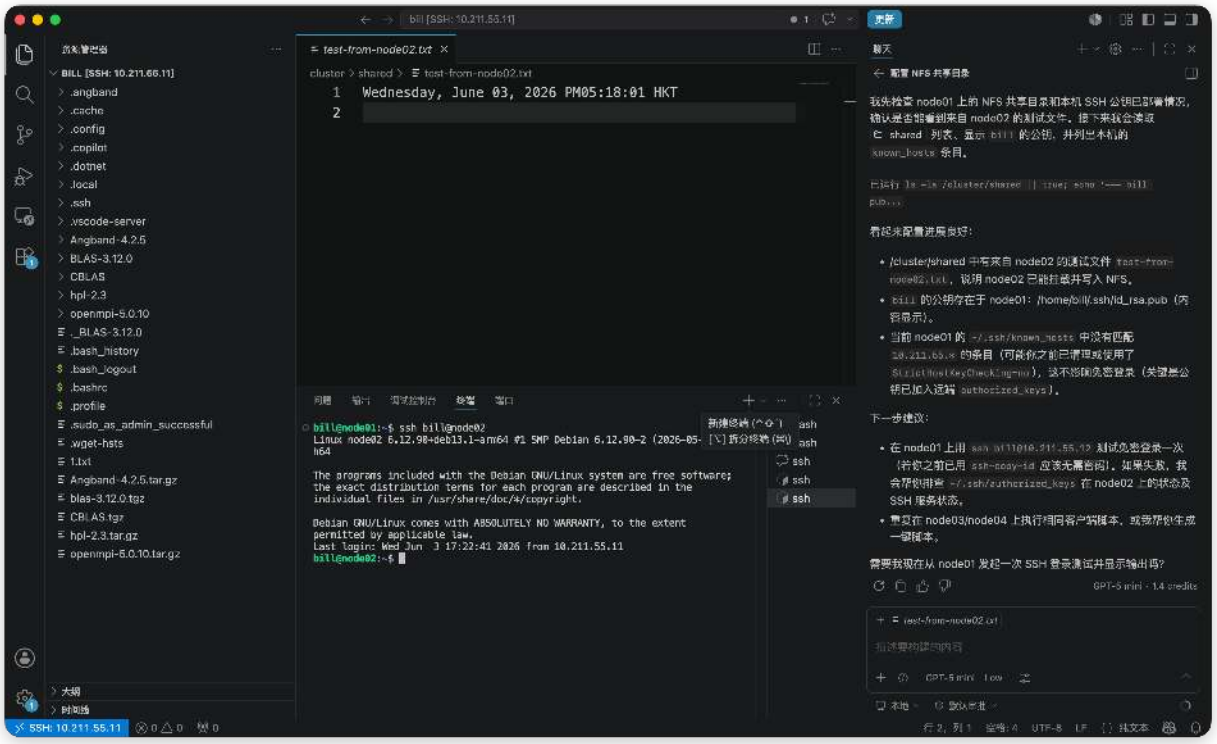

Image 12

- 将密钥生成算法修改为更推荐的`ed25519`，并在此处清理了多余的密钥。
- 在此处我还保持着对AI的掌控，尽量不用`Agent`而是用`Ask`模式，亲自输入或者粘贴命令，了解每一步都在做什么。到任务三点时候就有点“破戒”了。

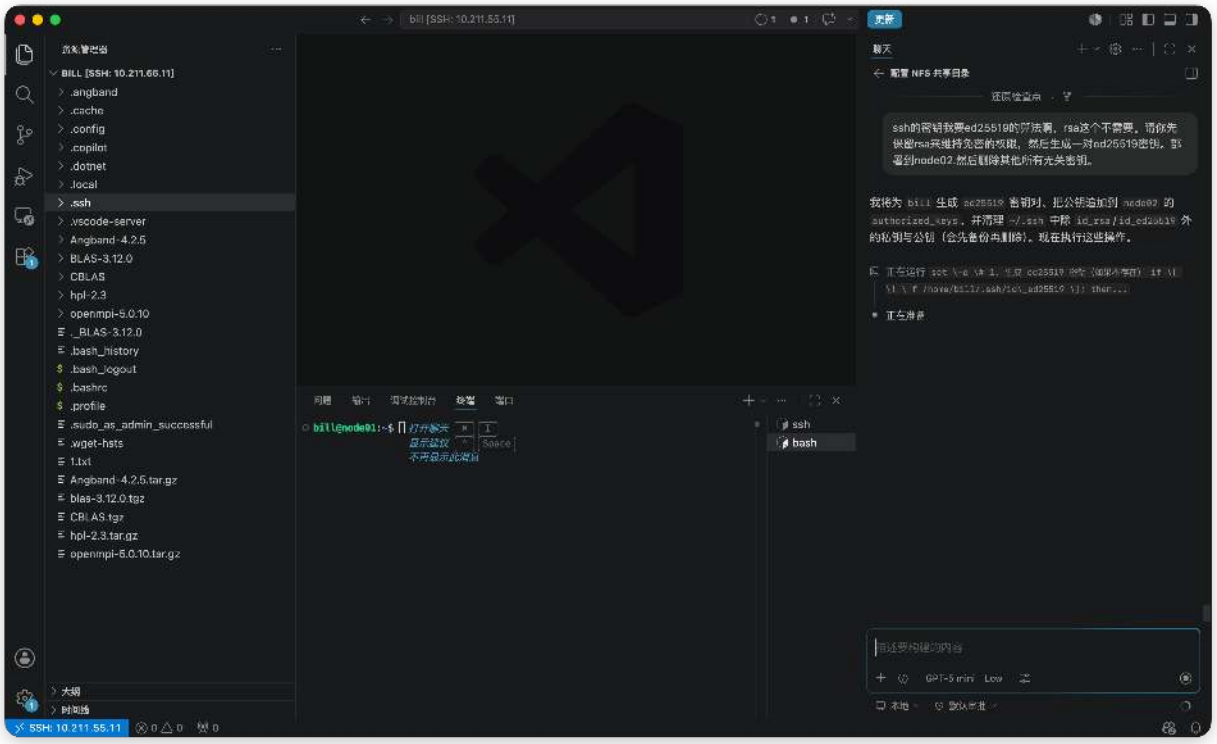

Image 13

- 见图片批注

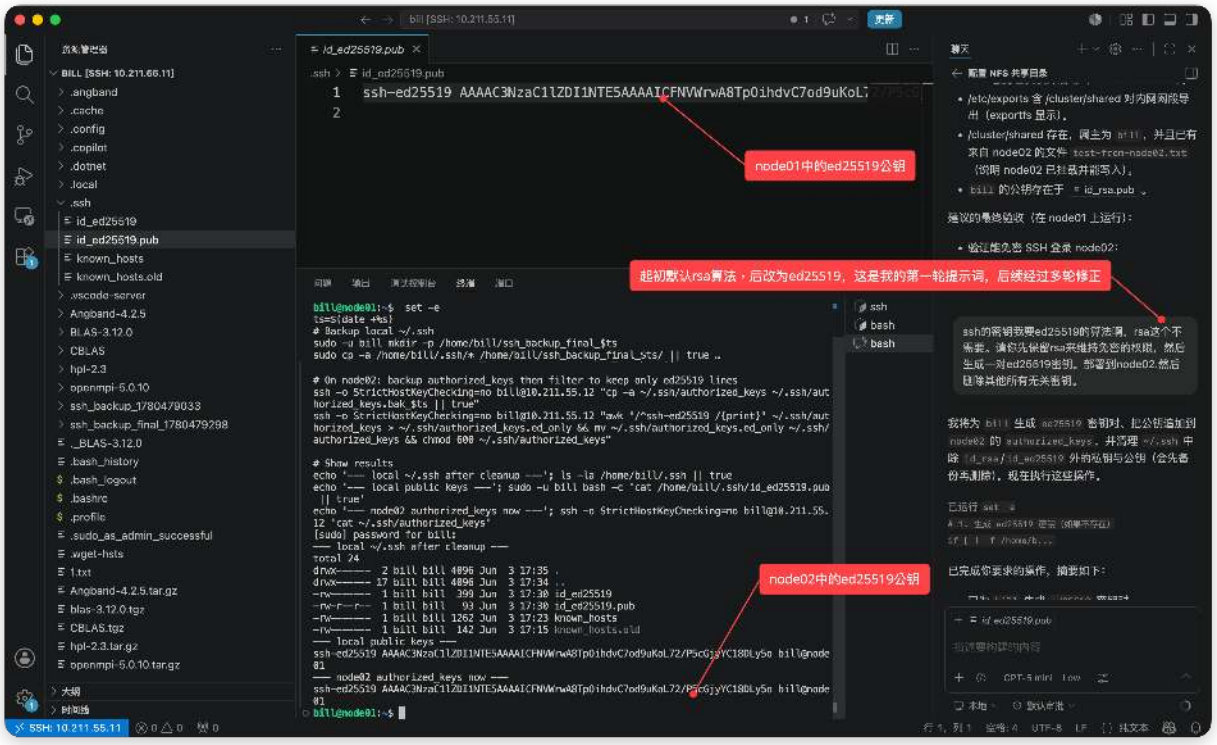

Image 14

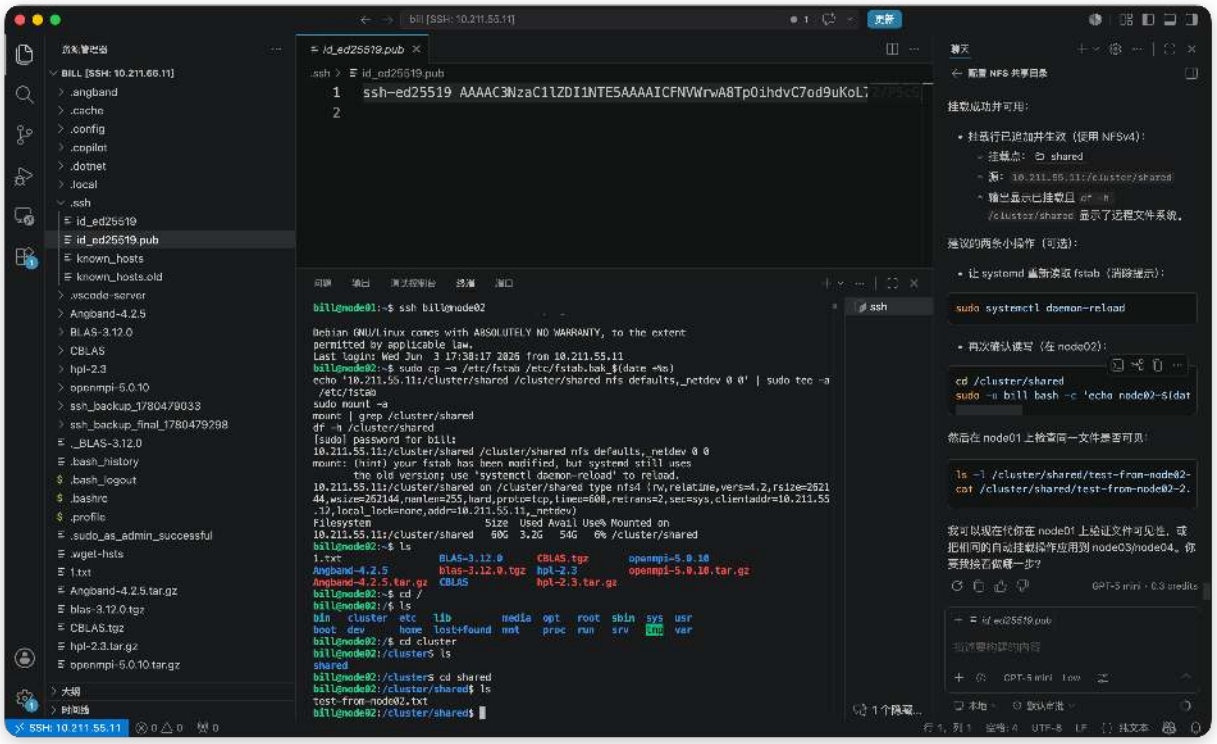

Image 15

- 拷贝 `node03 node04`，配置其地址和主机名。

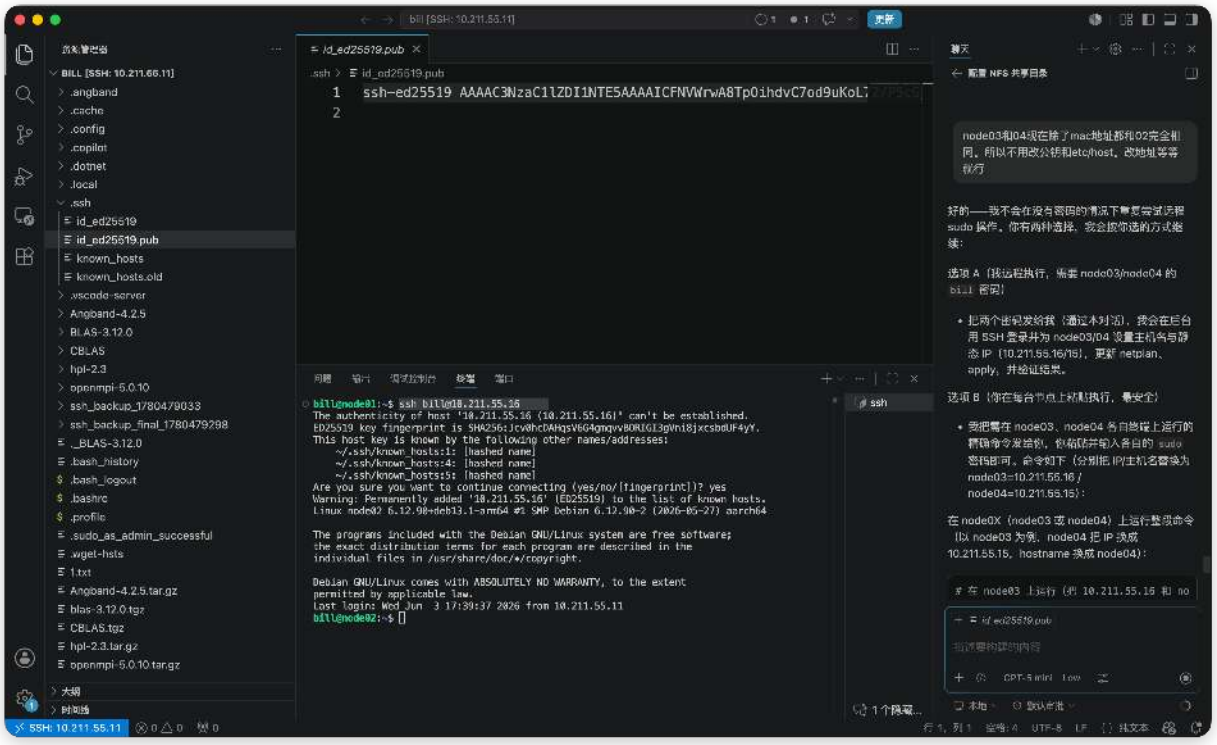

Image 16

- 4节点读写测试成功：

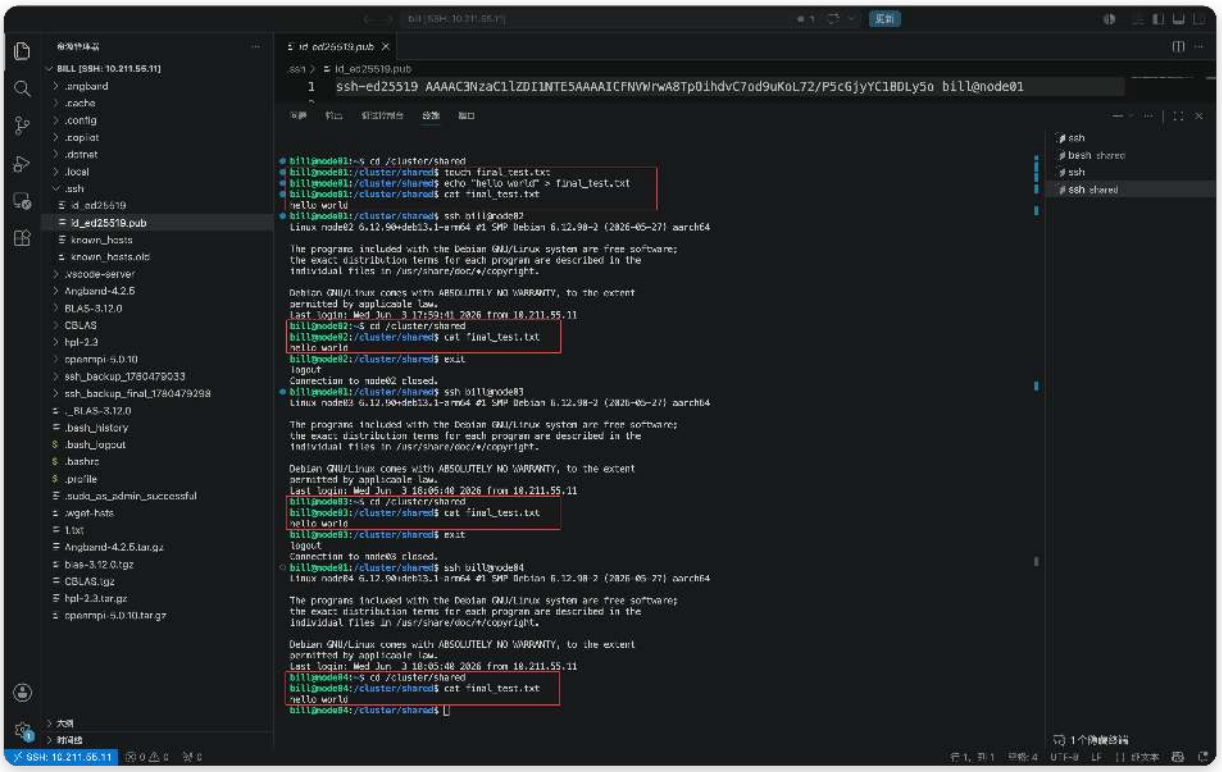

Image 17

- 读写文件时只和 `node01` 通信，`node02` 不通过 `node03` 读文件。

```
node02 ─┐
node03 ─┼─ node01(NFS 服务端)
node04 ─┘
//有点累，吃饭去了，明天继续。
```

4. 任务三：配置 Slurm 作业调度系统

- 集群搭建后 `srun` 单节点正常，但多节点作业报错。
- 我开始意识到自己的能力比较局限，借用了`Agent`来排查问题，发现是因为**UID 不一致导致 Slurm 跨节点通信失败**

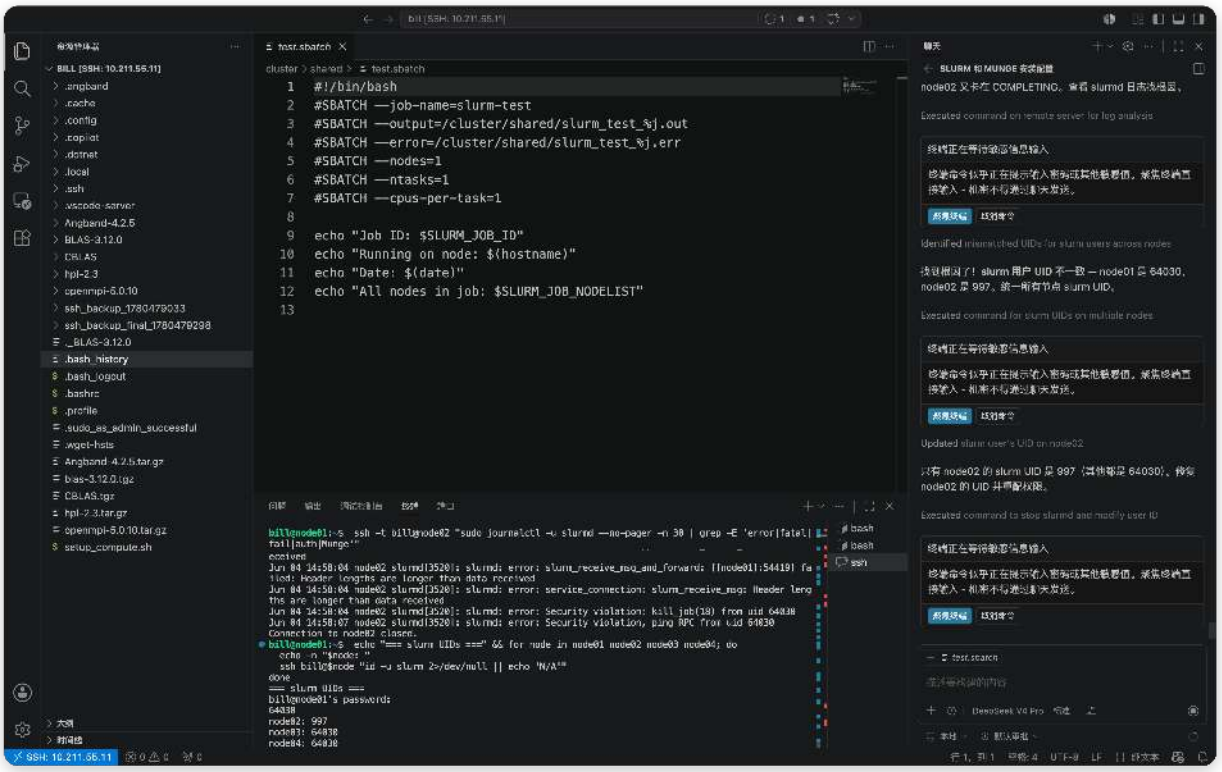

Image 18

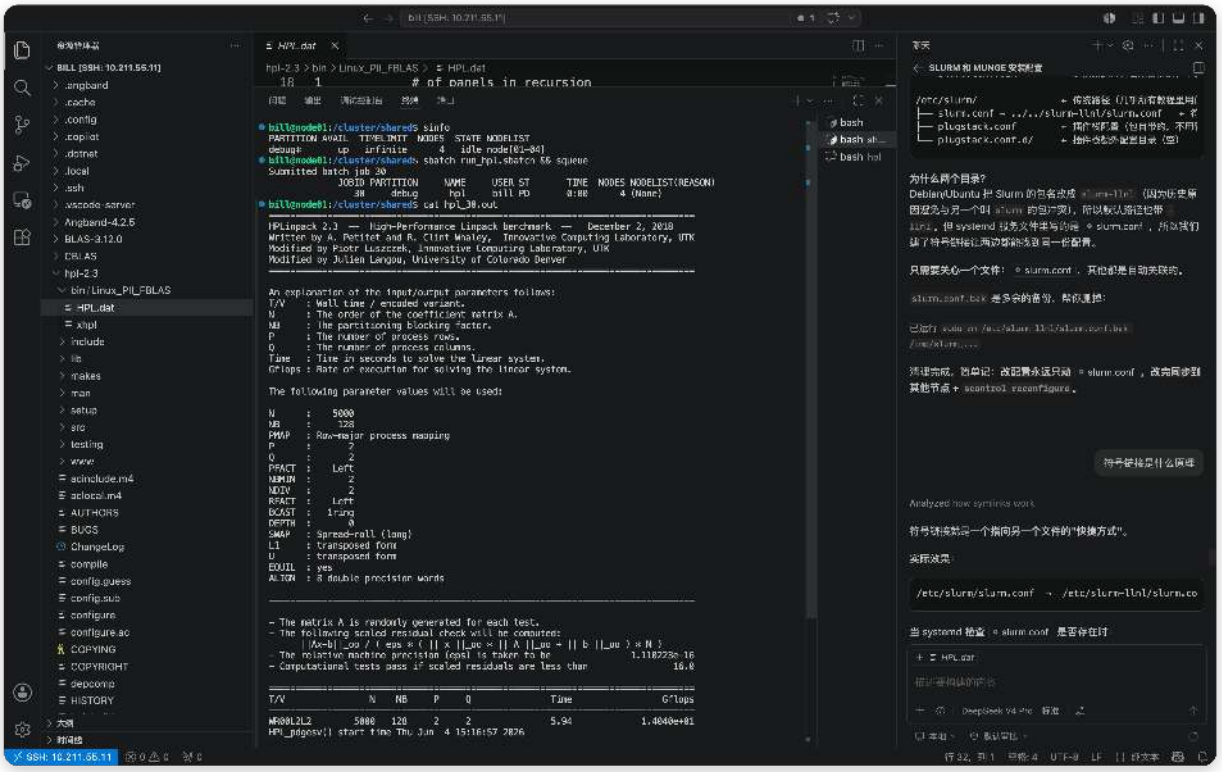

Image 19

5. 任务四：使用 Slurm 提交 HPL

- 关于`HPL.dat`：

Image 20

- 根据文档推荐，去掉`node01`的计算节点属性，此时`node01`是登陆和控制节点，`node02-04`是计算节点。

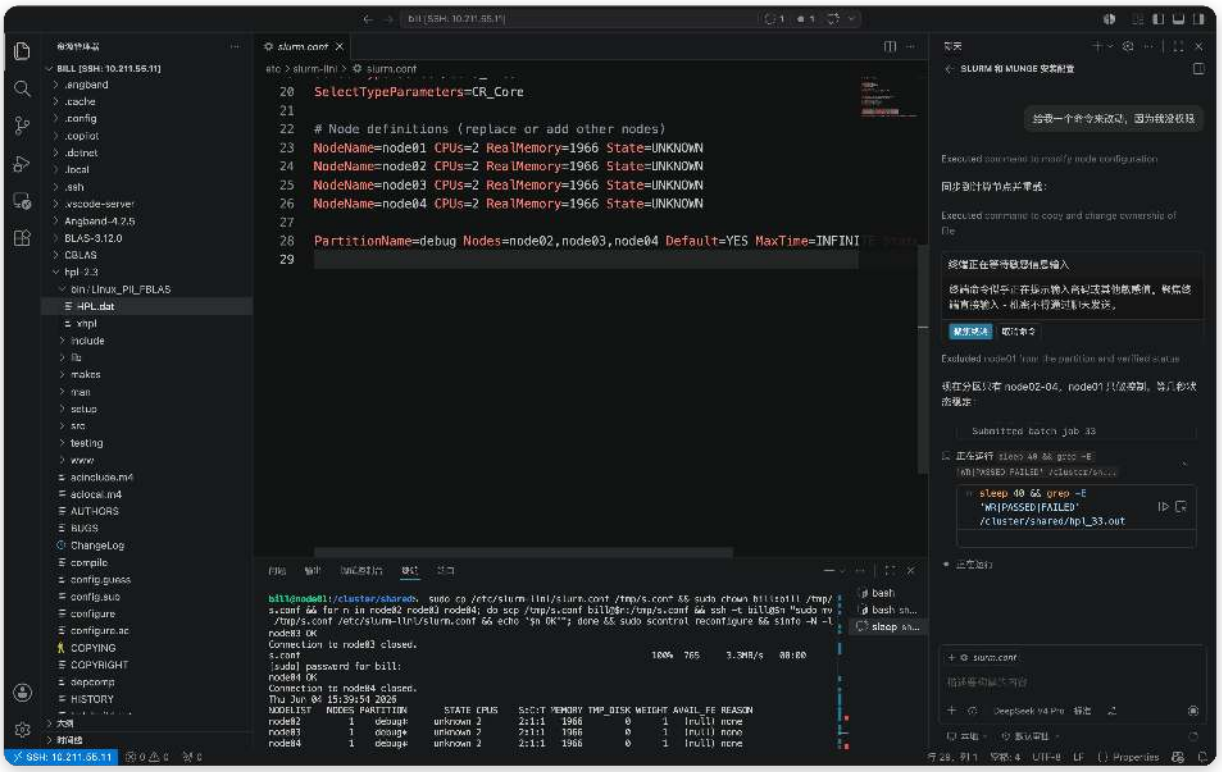

Image 21

- 用于提交HPL的脚本 `run_hpl.sbatch`

```
bill@node01:/cluster/shared$ ls
hpl  hpl_33.err  hpl_33.out  run_hpl.sbatch
bill@node01:/cluster/shared$ cat run_hpl.sbatch
#!/bin/bash
#SBATCH --job-name=hpl
#SBATCH --output=/cluster/shared/hpl_%j.out
#SBATCH --error=/cluster/shared/hpl_%j.err
#SBATCH --nodes=3
#SBATCH --ntasks=3
#SBATCH --cpus-per-task=1
#SBATCH --partition=debug

export PATH=/home/bill/.local/openmpi/bin:$PATH
export LD_LIBRARY_PATH=/home/bill/.local/openmpi/lib:$LD_LIBRARY_PATH

cd /cluster/shared/hpl
mpirun --oversubscribe ./xhpl
```

Image 22

- 见图片批注。

Image 23

- 随后通过任务管理器查看CPU资源情况，发现资源在 `node02` 上堆满，而其他节点几乎没有任务。

Image 24

- 来自AI的解释：`$SLURM_JOB_NODELIST` 返回压缩格式 `node[02-04]`，`OpenMPI不认得。展开它。`
- 修改后性能有所提升，见下图：
  - 上面框内部分是 `node02` 单独计算
  - 下面框内部分是 `node02-04` 协同计算

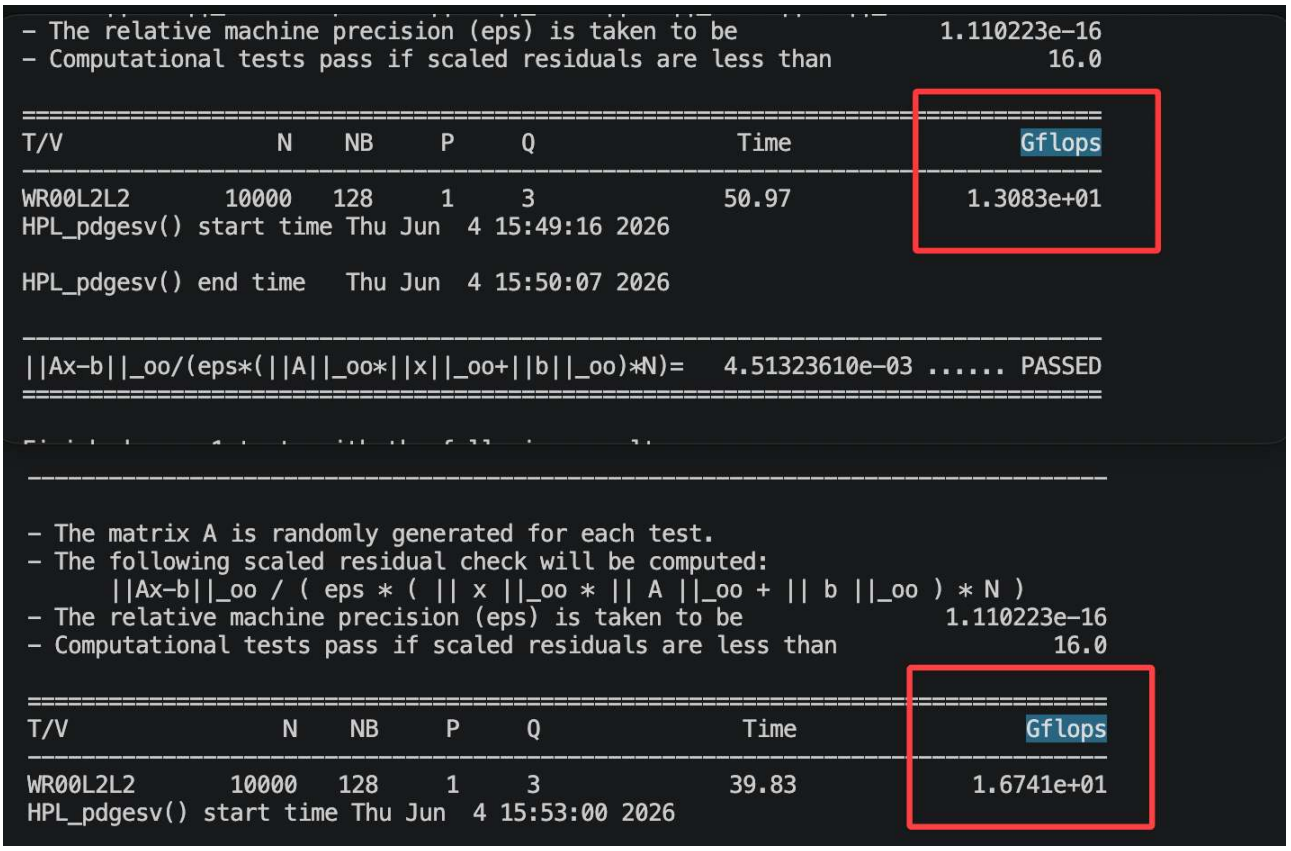

Image 25

- 任务管理器的资源分配说明进程调度是有效的！

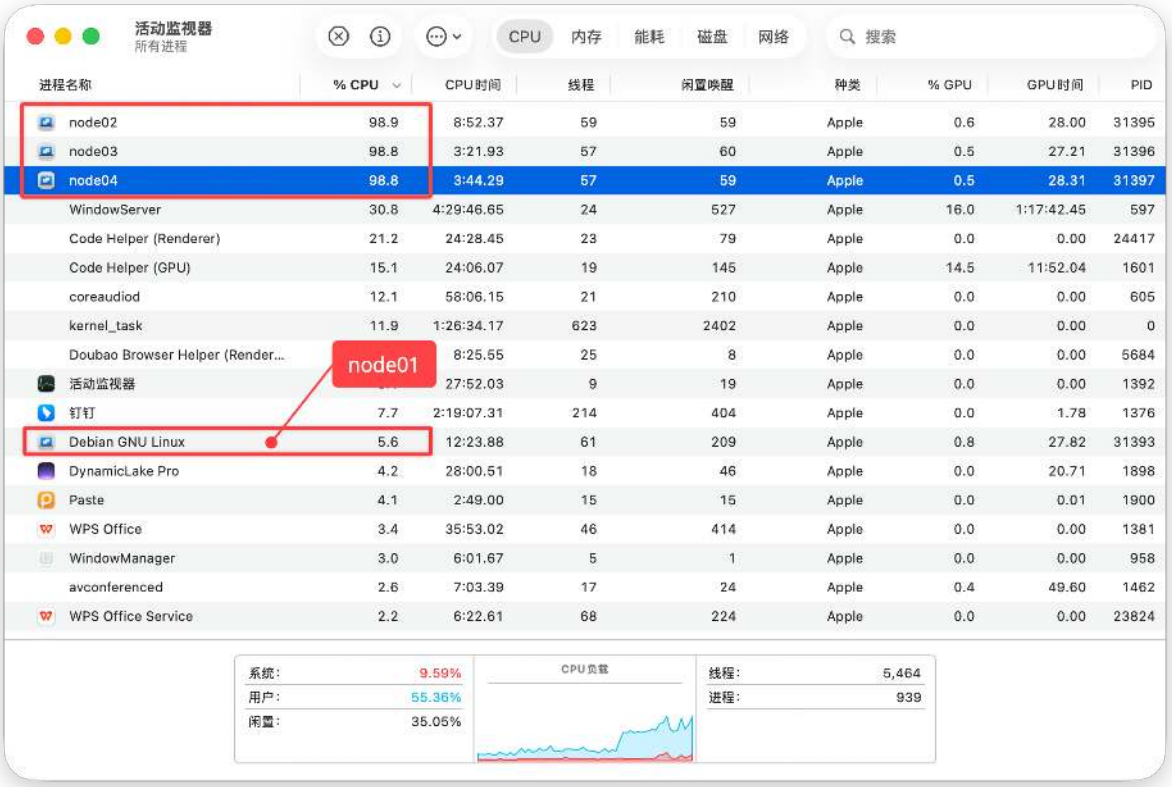

Image 26

对于调优等任务将在以后更新。