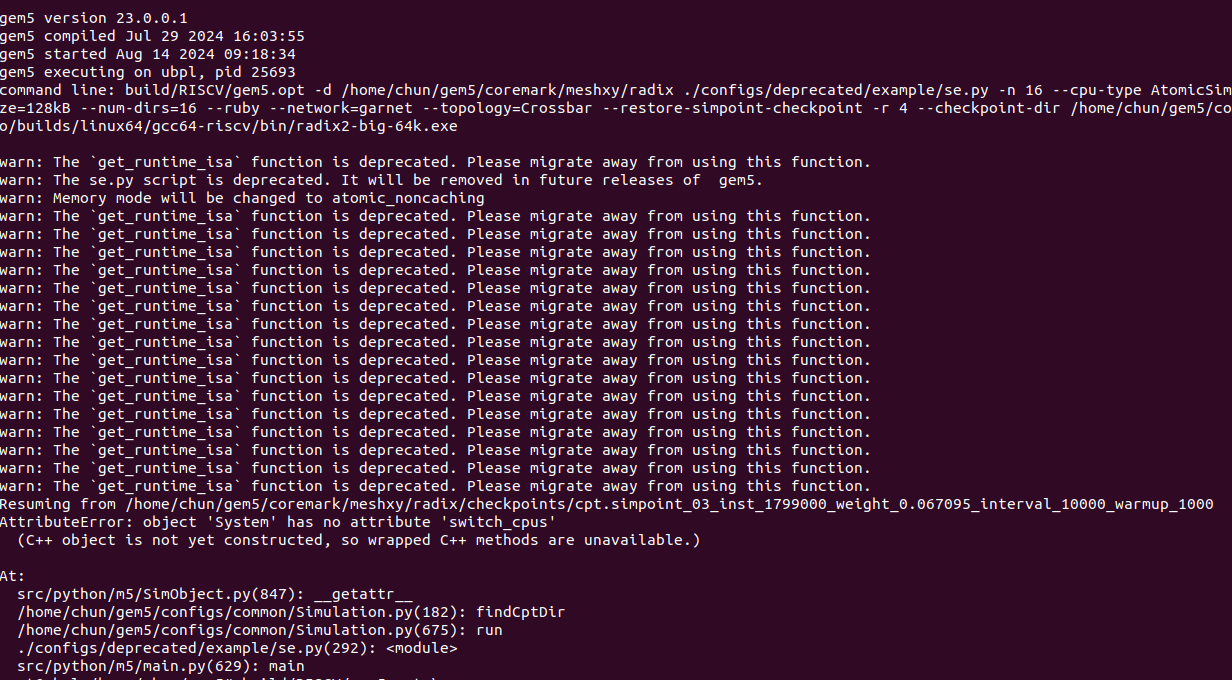
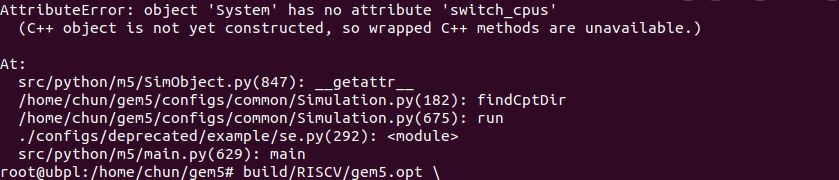
**不同CPU类型重载**

用AtomicSimpleCPU去重载TimingSimpleCPU生成的checkpoints，报错，no switch\_cpus

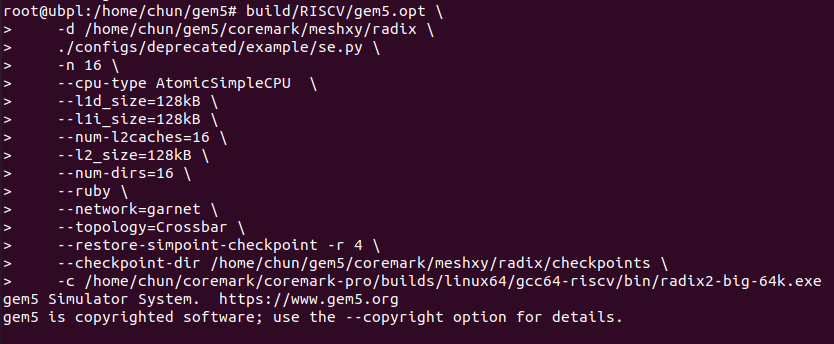


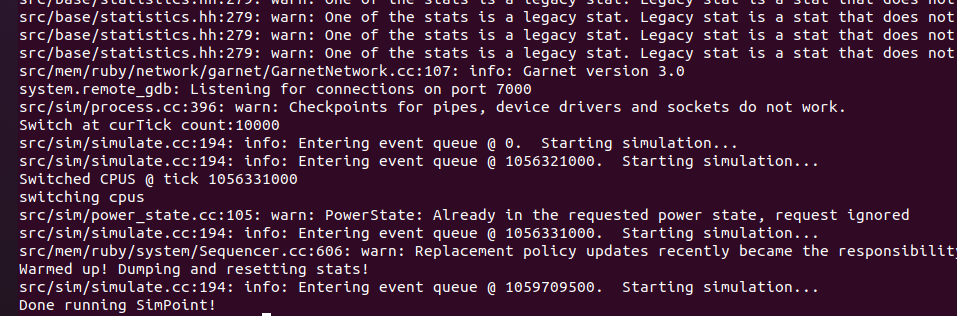


**同CPU类型，不同网络拓扑结构**

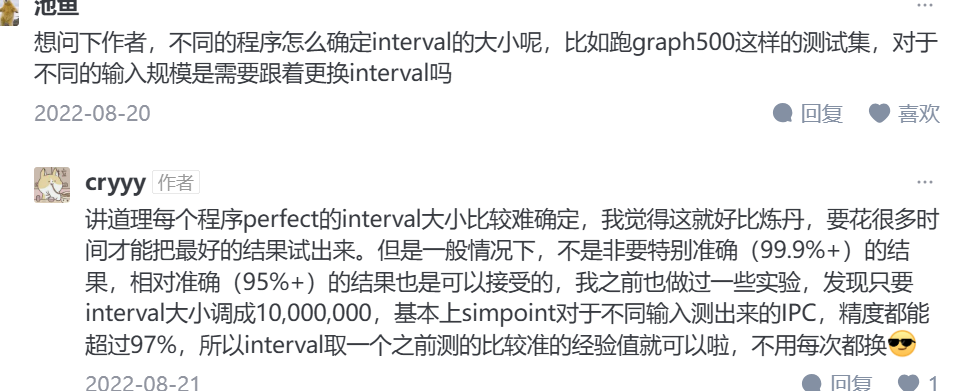
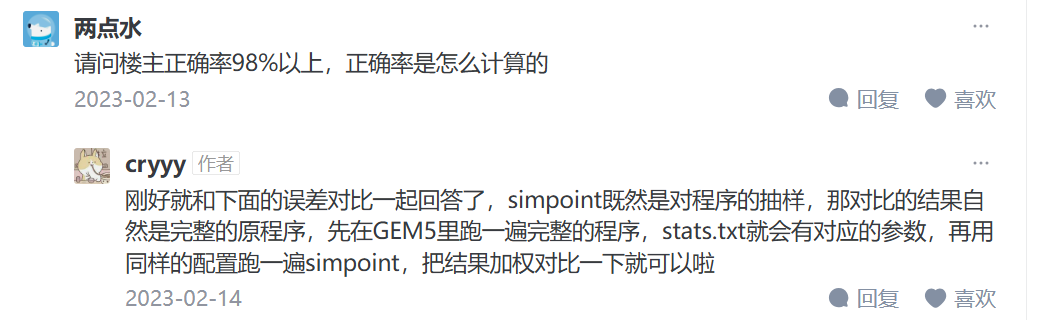
修改网络拓扑：

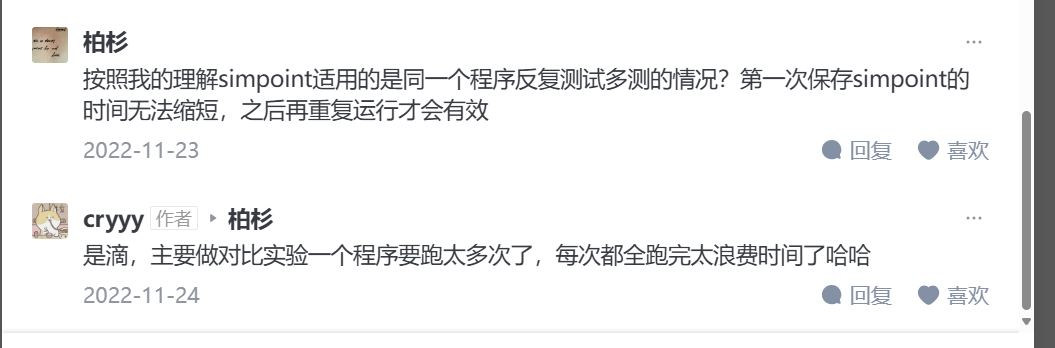
--network=garnet --topology=Mesh\_XY --mesh-rows=4 生成的checkpoints用另一个拓扑结构重载--network=garnet --topology=Crossbar 可以运行





**重载生成的结果统计问题**







CPI不是加权计算，重载的stats.txt里面有些数据和正常运行的很相似，有些不一样

**用github脚本统计重载生成的结果**

[Computer-Architecture-Learning/cal\_GEM5\_stats at main · cry-daniel/Computer-Architecture-Learning (github.com)](https://github.com/cry-daniel/Computer-Architecture-Learning/tree/main/cal_GEM5_stats)

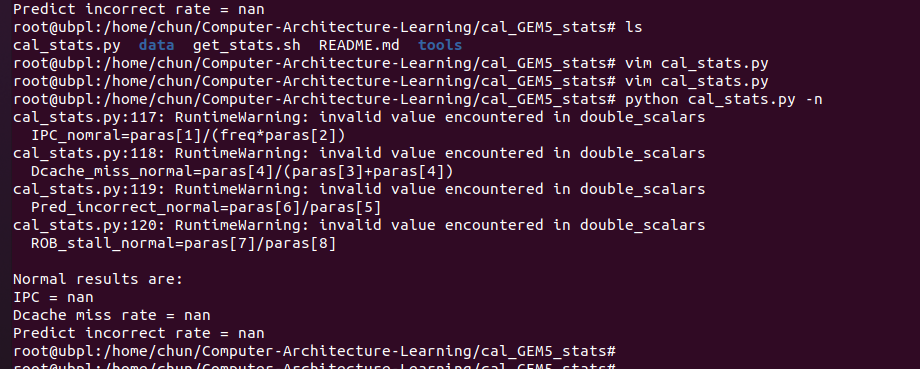
用github上的一个脚本来统计

IPC

Dcache miss rate

Predict incorrect rate

运行报错 结果为 nan

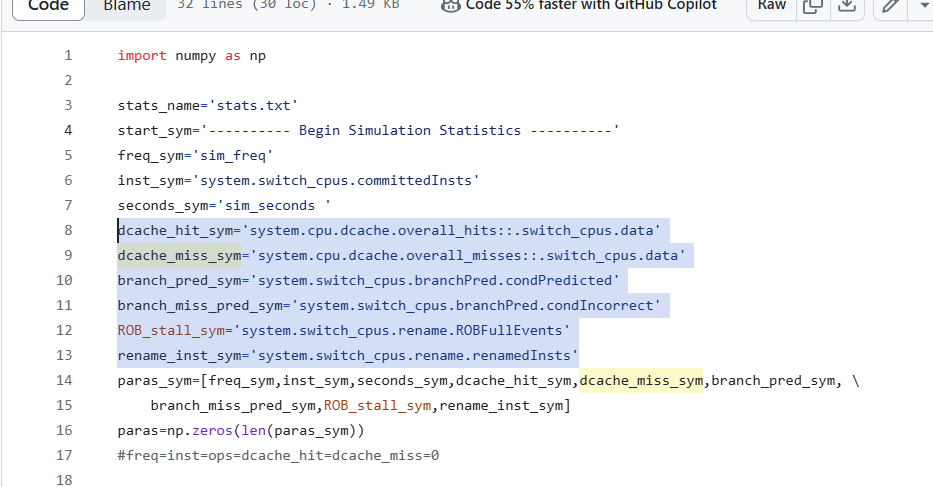


**不同的benchmark，生成的stats.txt里面的参数不一样**

后面看脚本代码 发现 不同的benchmark，生成的stats.txt里面的参数不一样，因为缓存、内存、网络拓扑、benchmark等等的不一样，stats.txt的统计结果也不同

不同benchmark生成的stats.txt文件不一样，蓝色框住的那6个参数stats.txt里面没有

Github统计脚本：



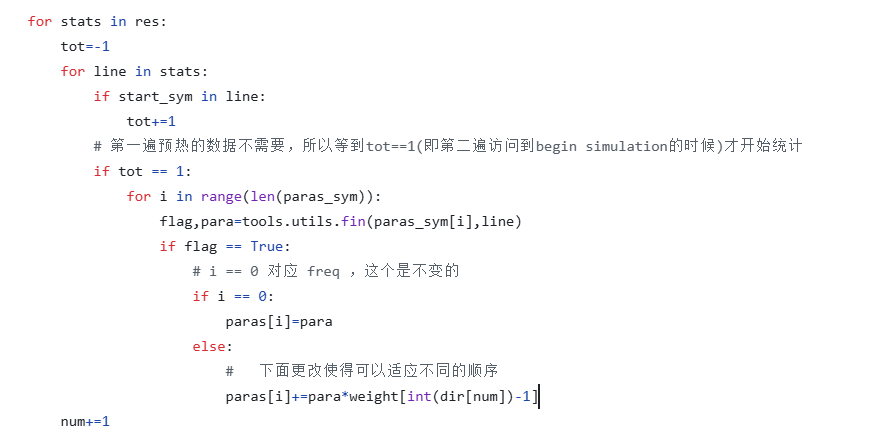
Github脚本是统计arm架构下graph500 benchmark 的，我自己生成的是RISCV架构下radix2-big-64k benchmark

不同配置参数，生成的stats.txt也会不同

所以每个不同的benchmark和配置参数不一样就要重写不同的统计脚本？

**不需要第一遍预热得数据！！**

Github统计脚本：

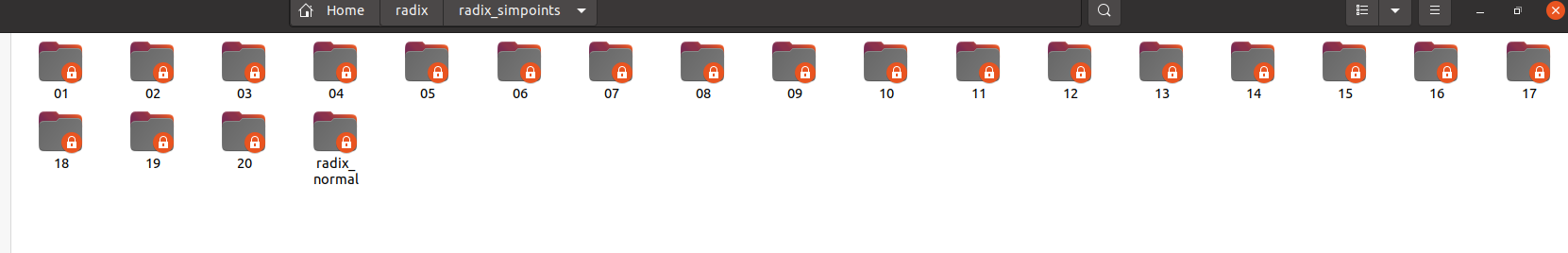


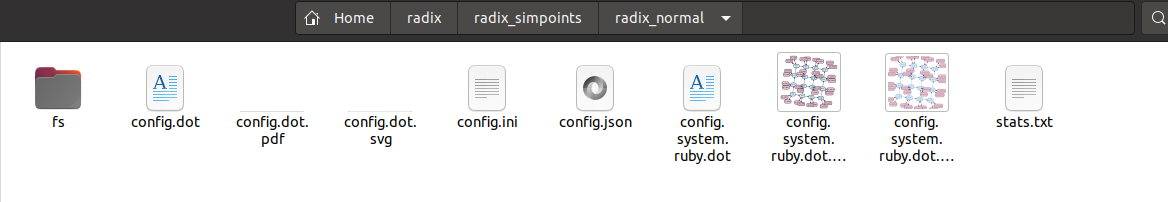
观察github统计stats.txt的脚本发现， 他统计的时候略过了第一遍访问begin simulation的数据，即第一遍预热的数据，难怪前面的数据统计起来差别很大。

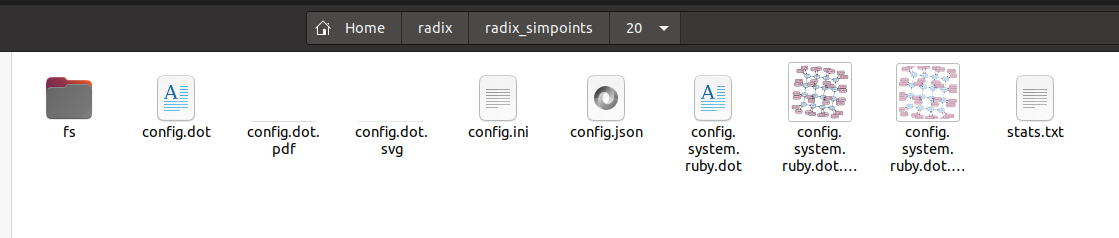
预热即重载时指定的warm up 参数

**手动统计：**

01-20为20个checkpoint重载的输出，radix\_normal为正常运行的输出：



Normal：

第20个checkpoints重载输出：

没弄清楚stats.txt里面一些参数的计算方法？

有些参数很相似，有些不一样

01：

|  |  |
| --- | --- |
| simSeconds | 1.8E-05 |
| simTicks | 17696500 |
| finalTick | 30166000 |
| simFreq | 1000000000000 |
| hostSeconds | 0.16 |
| hostTickRate | 108184972 |
| hostMemory | 759848 |
| simInsts | 11015 |
| simOps | 11019 |
| hostInstRate | 67307 |
| hostOpRate | 67329 |
| system.clk\_domain.clock | 1000 |

Normal：

|  |  |
| --- | --- |
| simSeconds | 0.182491 |
| simTicks | 182491096000 |
| finalTick | 182491096000 |
| simFreq | 1000000000000 |
| hostSeconds | 273.57 |
| hostTickRate | 667080848 |
| hostMemory | 826916 |
| simInsts | 102979391 |
| simOps | 102979531 |
| hostInstRate | 376432 |
| hostOpRate | 376433 |
| system.clk\_domain.clock | 1000 |

**system.cpu0.cpi**

Normal ：system.cpu0.cpi 24.005996

0：1.333333

1：1.052632

2：1.052632

3： 1.111111

4：1.111111

5：1.176471

6：1.052632

7： 1.111111

8： 1.111111

9： 1.176471

10： 1.111111

11：1.105263

12：1.111111

13： 1.166667

14：1.235294

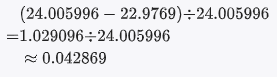
15： 1.111111

16：1.111111

17： 1.111111

18： 1.250000

19： 1.111111



0.0428误差

**system.cpu0.ipc**

0： 0.750000\*0.0435403

1：0.950000\*0.0328337

2：0.950000\*0.00499643

3： 0.900000\*0.0421128

4： 0.900000\*0.0756602

5：0.850000\*0.713776

6：0.950000\*0.0992148

7： 0.900000\*0.0670949

8： 0.900000\*0.0820842

9： 0.850000\*0.0513919

10：0.900000\*0.0706638

11： 0.904762\*0.082798

12：0.900000\*0.0606709

13： 0.857143\*0.000713776

14：0.809524\*0.06995

15： 0.900000\*0.0842256

16：0.900000\*0.034975

17： 0.900000\*0.0185582

18：0.800000\*0.00499643

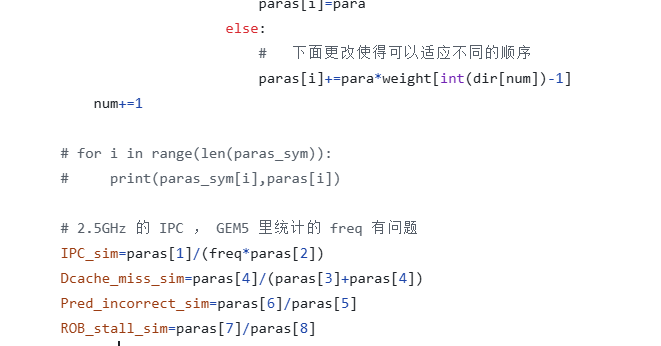
19： 0.900000\*0.00214133

Normal： 0.038453

发现算出来的误差太大 ，应该是IPC的计算方法不对？

github脚本里IPC的计算如下

统计脚本：



可以看到stats.txt里的参数计算方法是：para\*权重再累加

即：

para = 0.22 ∗ para1 + 0.33 ∗ para2 + 0.45 ∗ para3

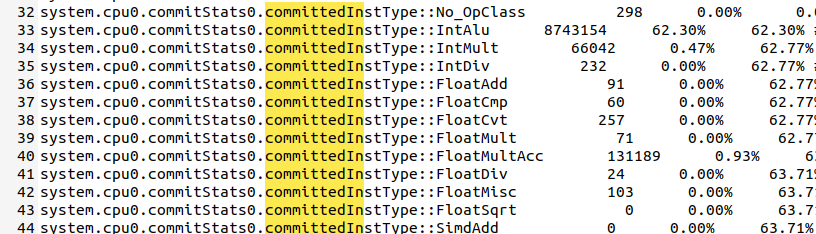
但每个指标的计算方法不一样，

IPC=system.switch\_cpus.committedInsts/sim\_freq\*sim\_seconds

Dcache和pre\_incorrect 这两个指标里面的参数在radix生成的stats.txt下没有

我生成的stats.txt里面没有system.switch\_cpus.committedInsts 这个参数，只有committedInstType

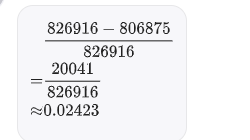
RISCV架构下radix benchmark生成的stats.txt：



**手动统计：**

重载加权和hostMemory = 759316 \* 0.0142857+ 761112 \*0.071428+ ........ =806875.693252

正常运行hostMemory = 826916



hostMemory参数的误差为2.423%

问题：

一个stats.txt里面的参数太多了 1w多条参数，哪些是需要的哪些是不需要的？不同benchmark具体要统计哪些指标？

不同benchmark生成的stats还不一样，

不同benchmark 所统计的指标也不同，graph500没用到网络拓扑，stats.txt里就没有网络拓扑相关的参数，

radix 的stats.txt里面大多是网络拓扑相关的参数。

目前进展：

网上暂时只找到这一个统计gem5 stat.txt的脚本，现在是在琢磨怎么自己重写一个脚本来统计，但好像每个不同benchmark就要写不同的脚本。