PoW 及其攻击模拟

目录树

```
src
│   attack.py
│   fork.py
│   pow.py
│
├──data
│   ├──task_1
│   ├──task_2
│   └──task_3
└──figures
```

PoW 程序设计

每个结点与两个线程相对应，其中一个线程（mine）负责挖矿，另一个线程（receive）负责接收消息。

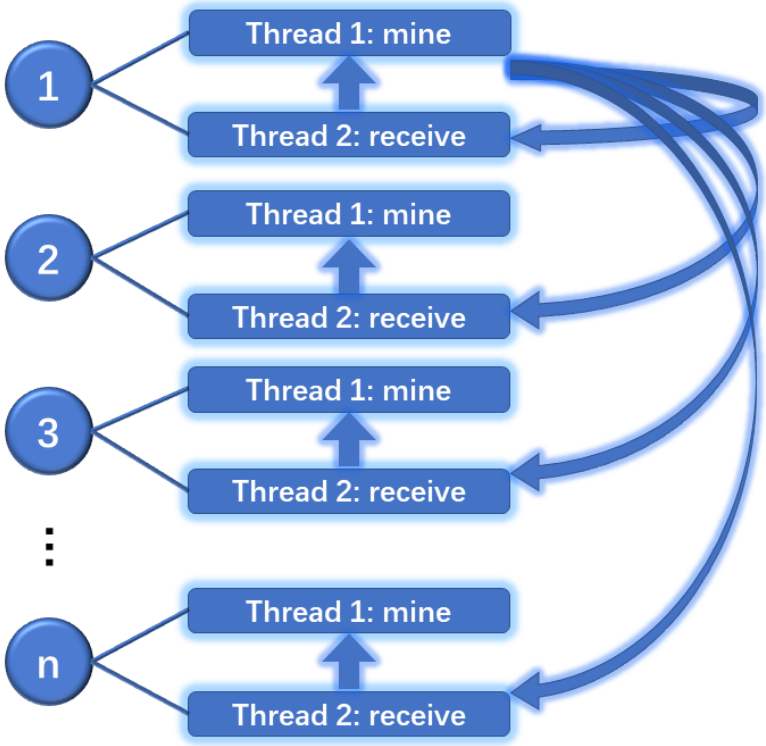
每个结点维护两条链，一条是达成共识的链，在结点之间通信时进行更新；另一条是挖矿的链，当结点在共识链的基础上挖出一个新块时，将新链赋值给挖矿的链，然后对挖矿的链进行广播。

在广播时，将一个结点的挖矿链放到所有结点的消息队列（channel）里，然后通知所有结点的接收消息的线程，进行链更新。

接收消息的链收到消息后，如果发现它所在结点的共识链长度小于接收到消息中的链的最大值，就更新这条共识链，并通知自己结点的挖矿链停止挖矿，在新的共识链的基础上接着挖。

PoW 模拟程序需要用到 Python threading 库中的 Event 和 Lock 两个类，前者负责线程之间的通信和共识，后者负责对消息队列这一全局变量实现互斥操作。

PoW 模拟程序的工作原理如下图所示。



分叉攻击程序设计

分叉攻击与自私挖矿较为相似，我设计了简单的模拟程序。将所有结点分为攻击结点和诚实结点，分别在各自范围内进行内部通信。为简单起见，二者共同维护创世块，其下一个结点就开始分叉。将出块和共识两种消息写入输出文件。

自私挖矿程序设计

另外设置一个全局布尔变量，表示自私挖矿是否开始。

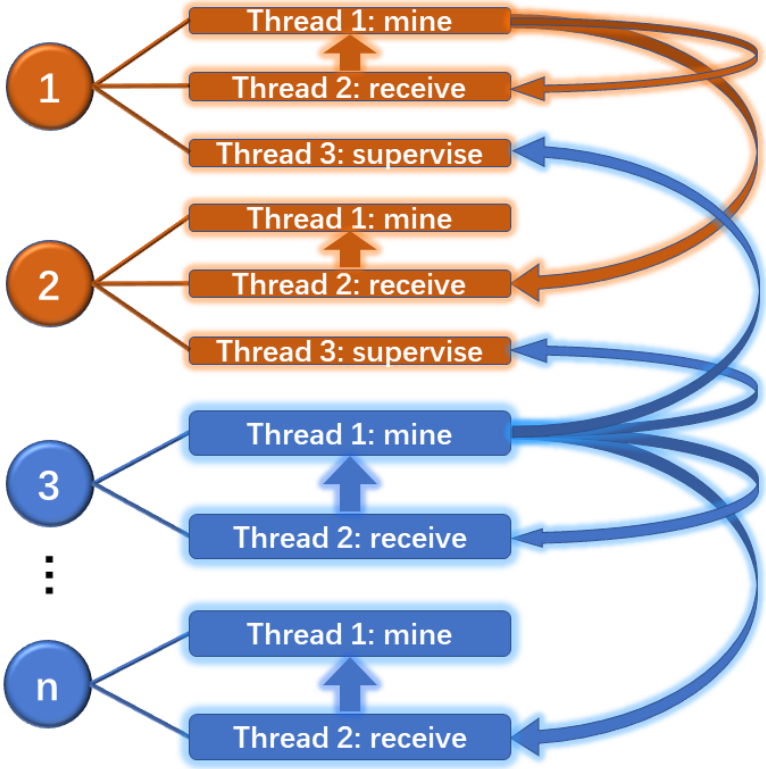
自私挖矿开始的标志是某一个攻击结点在共识链的基础上挖出了一个新块（只有最先挖出新块的结点会启动自私挖矿，因为它会在出块的同时通知其他没出块的结点停止当前的挖矿）。此时自私挖矿开始生效，具体包括：

- 攻击结点内部通信，诚实结点无法得到攻击结点挖出的块
- 诚实结点仍将自己的消息进行全网广播
- 当攻击结点发现自己内部维护的共识链长度超过了诚实结点所维护的共识链，就停止自私挖矿，将挖到的长链广播出去。

为了便于实现，我在代码中为攻击结点开设了第三个线程，名为 supervise。相应地也开设了第二个消息队列（sv_channel）。当自私挖矿开始时，结点与线程之间的操作如下：

- 攻击结点在挖出新块后，将挖矿链发给所有攻击结点的 receive 线程（通过 channel）
- 诚实结点在挖出新块后，将挖矿链发给所有诚实结点的 receive 线程（通过 channel）
- 诚实结点在达成一次共识后，将共识链发给所有攻击结点的 supervise 线程（通过 sv_channel）

自私挖矿模拟程序的工作原理如下图所示，在程序运行时，恶意结点一旦有机会就马上发起攻击，当发现自己内部所维护的链比公开的链长时，就马上将其发布出去。



参数设置

PoW 模拟程序

参数包括结点数、挖矿的 hash_difficulty（即要求哈希值的前多少个比特位是 0）和最大链长。实验中结点数设为 20，hash_difficulty 设为 8, 12, 16，最大链长设为 50

experiment	nodes	hash difficulty	length of chain
PoW Simulator	20	8, 12, 16	50

分叉攻击模拟程序

参数在 PoW 模拟程序的基础上，还包括攻击结点数。实验中结点数设为 10，hash_difficulty 设为 12，最大链长设为 20，攻击结点比例从 10% 到 40% 分别测试。

experiment	nodes	hash difficulty	length of chain	attack percentage
Selfish Mining Simulator	10	12	20	10%, 20%, 30%, 40%

自私挖矿模拟程序

参数在 PoW 模拟程序的基础上，还包括攻击结点数。实验中结点数设为 10，hash_difficulty 设为 12，最大链长设为 40，攻击结点比例从 10% 到 40% 分别测试。

experiment	nodes	hash difficulty	length of chain	attack percentage
Selfish Mining Simulator	10	12	40	10%, 20%, 30%, 40%

实验结果

PoW 模拟程序

实验中输出每一个结点所维护的共识链，并计算达成共识的区块总数。在不同的 hash difficulty 参数设定下，第一个结点所维护的链可视化如下：

1. hash difficulty = 8，47 / 50 个块达成共识，平均出块时间为 0.37 s

```
Node 0, Block 0, Nounce 0, Source Id -1, time stamp
Node 0, Block 1, Nounce 56, Source Id 0, time stamp 2023-01-08 19:26:47.763161
Node 0, Block 2, Nounce 633, Source Id 0, time stamp 2023-01-08 19:26:47.763771
Node 0, Block 3, Nounce 536, Source Id 0, time stamp 2023-01-08 19:26:47.787916
Node 0, Block 4, Nounce 188, Source Id 0, time stamp 2023-01-08 19:26:47.804805
Node 0, Block 5, Nounce 565, Source Id 0, time stamp 2023-01-08 19:26:47.864841
Node 0, Block 6, Nounce 541, Source Id 0, time stamp 2023-01-08 19:26:47.959056
Node 0, Block 7, Nounce 272, Source Id 0, time stamp 2023-01-08 19:26:48.201294
Node 0, Block 8, Nounce 299, Source Id 1, time stamp 2023-01-08 19:26:48.264821
Node 0, Block 9, Nounce 602, Source Id 1, time stamp 2023-01-08 19:26:48.445677
Node 0, Block 10, Nounce 27, Source Id 8, time stamp 2023-01-08 19:26:48.543676
Node 0, Block 11, Nounce 79, Source Id 8, time stamp 2023-01-08 19:26:49.386373
Node 0, Block 12, Nounce 152, Source Id 12, time stamp 2023-01-08 19:26:49.423367
Node 0, Block 13, Nounce 98, Source Id 12, time stamp 2023-01-08 19:26:49.706313
Node 0, Block 14, Nounce 301, Source Id 1, time stamp 2023-01-08 19:26:49.747162
Node 0, Block 15, Nounce 411, Source Id 12, time stamp 2023-01-08 19:26:50.777409
Node 0, Block 16, Nounce 237, Source Id 12, time stamp 2023-01-08 19:26:51.567408
Node 0, Block 17, Nounce 259, Source Id 2, time stamp 2023-01-08 19:26:51.574687
Node 0, Block 18, Nounce 347, Source Id 2, time stamp 2023-01-08 19:26:52.611245
Node 0, Block 19, Nounce 196, Source Id 1, time stamp 2023-01-08 19:26:53.791137
```

2. hash difficulty = 12，48 / 50 个块达成共识，平均出块时间为 1.11 s

```
Node 0, Block 0, Nounce 0, Source Id -1, time stamp
Node 0, Block 1, Nounce 4962, Source Id 1, time stamp 2023-01-07 22:28:45.854920
Node 0, Block 2, Nounce 430, Source Id 1, time stamp 2023-01-07 22:28:45.889961
Node 0, Block 3, Nounce 340, Source Id 1, time stamp 2023-01-07 22:28:45.902639
Node 0, Block 4, Nounce 8519, Source Id 1, time stamp 2023-01-07 22:28:45.920183
Node 0, Block 5, Nounce 199, Source Id 0, time stamp 2023-01-07 22:28:46.042187
Node 0, Block 6, Nounce 5584, Source Id 0, time stamp 2023-01-07 22:28:46.180373
Node 0, Block 7, Nounce 1225, Source Id 0, time stamp 2023-01-07 22:28:46.353350
Node 0, Block 8, Nounce 1832, Source Id 0, time stamp 2023-01-07 22:28:46.556727
Node 0, Block 9, Nounce 1378, Source Id 0, time stamp 2023-01-07 22:28:46.901757
Node 0, Block 10, Nounce 1670, Source Id 9, time stamp 2023-01-07 22:28:47.205723
Node 0, Block 11, Nounce 804, Source Id 9, time stamp 2023-01-07 22:28:47.464758
Node 0, Block 12, Nounce 6957, Source Id 4, time stamp 2023-01-07 22:28:47.933466
Node 0, Block 13, Nounce 2072, Source Id 4, time stamp 2023-01-07 22:28:48.567342
Node 0, Block 14, Nounce 5392, Source Id 4, time stamp 2023-01-07 22:28:49.485607
Node 0, Block 15, Nounce 4716, Source Id 4, time stamp 2023-01-07 22:28:50.082401
Node 0, Block 16, Nounce 700, Source Id 4, time stamp 2023-01-07 22:28:50.498962
Node 0, Block 17, Nounce 4084, Source Id 4, time stamp 2023-01-07 22:28:51.122701
Node 0, Block 18, Nounce 3495, Source Id 4, time stamp 2023-01-07 22:28:51.779660
Node 0, Block 19, Nounce 3120, Source Id 3, time stamp 2023-01-07 22:28:52.225020
```

3. hash difficulty = 16，47 / 50 个块达成共识，平均出块时间为 6.06 s

```
Node 0, Block 0, Nounce 0, Source Id -1, time stamp
Node 0, Block 1, Nounce 5128, Source Id 3, time stamp 2023-01-08 19:35:37.912445
Node 0, Block 2, Nounce 4720, Source Id 3, time stamp 2023-01-08 19:35:38.191410
Node 0, Block 3, Nounce 6823, Source Id 3, time stamp 2023-01-08 19:35:39.907371
Node 0, Block 4, Nounce 3064, Source Id 8, time stamp 2023-01-08 19:35:41.003167
Node 0, Block 5, Nounce 7156, Source Id 8, time stamp 2023-01-08 19:35:46.687996
Node 0, Block 6, Nounce 6296, Source Id 0, time stamp 2023-01-08 19:35:52.363527
Node 0, Block 7, Nounce 7492, Source Id 0, time stamp 2023-01-08 19:35:58.051807
Node 0, Block 8, Nounce 4881, Source Id 1, time stamp 2023-01-08 19:36:03.894781
Node 0, Block 9, Nounce 300, Source Id 1, time stamp 2023-01-08 19:36:10.716883
Node 0, Block 10, Nounce 7236, Source Id 19, time stamp 2023-01-08 19:36:15.897289
Node 0, Block 11, Nounce 1868, Source Id 19, time stamp 2023-01-08 19:36:21.730591
Node 0, Block 12, Nounce 7626, Source Id 5, time stamp 2023-01-08 19:36:22.853707
Node 0, Block 13, Nounce 8940, Source Id 5, time stamp 2023-01-08 19:36:36.385007
Node 0, Block 14, Nounce 2012, Source Id 15, time stamp 2023-01-08 19:36:44.169591
Node 0, Block 15, Nounce 9998, Source Id 15, time stamp 2023-01-08 19:36:47.839361
Node 0, Block 16, Nounce 2597, Source Id 3, time stamp 2023-01-08 19:36:54.608835
Node 0, Block 17, Nounce 6777, Source Id 3, time stamp 2023-01-08 19:36:59.627950
Node 0, Block 18, Nounce 8218, Source Id 1, time stamp 2023-01-08 19:37:01.702510
Node 0, Block 19, Nounce 9987, Source Id 9, time stamp 2023-01-08 19:37:09.870635
```

分叉攻击模拟程序

在分叉攻击模拟程序的参数设定下，进行 4 组实验，每组重复 3 次，每次取最先达成共识的 10 个区块，统计在诚实链和攻击链中各占多少：

恶意结点比例	实验一（攻击链/诚实链）		实验二（攻击链/诚实链）		实验三（攻击链/诚实链）	
10%	6	4	5	5	7	3
20%	6	4	4	6	5	5
30%	5	5	7	3	5	5
40%	6	4	6	4	6	4

每次取最先达成共识的 20 个区块，统计在诚实链和攻击链中各占多少：

恶意结点比例	实验一（攻击链/诚实链）		实验二（攻击链/诚实链）		实验三（攻击链/诚实链）	
10%	7	13	8	12	8	12
20%	9	11	11	9	11	9
30%	8	12	12	8	11	9

恶意结点比例	实验一（攻击链/诚实链）		实验二（攻击链/诚实链）		实验三（攻击链/诚实链）	
40%	10	10	10	10	11	9

从以上统计结果可以看出，当恶意结点比例较低时，随着链的增长，刚一开始攻击可能奏效，但往后诚实链会占优。如果恶意结点比例较高，随着链的增长攻击有更大的概率成功。在实际系统中可以一段一段地发起分叉攻击，攻击结点如果发现自己维护的链比诚实链更长，就马上公布出去，见好就收，当日后有机会时，再发起新一轮攻击。

自私挖矿模拟程序

在自私挖矿模拟程序参数的设定下，进行 4 次实验，结果如下：

恶意结点比例	自私挖矿收益比例	诚实挖矿收益比例	自私挖矿成功链长/达成共识的链长
10%	2.63%	97.37%	1/38
20%	18.92%	81.08%	7/37
30%	25.71%	74.29%	9/35
40%	68.42%	31.58%	26/38

实验结果与预期相符，即当恶意结点比例超过 1/3 时，自私挖矿可以额外获利。