EE 374: Internet-Scale Consensus in the Blockchain Era **Stanford, Winter 2021**

Lecture 4: Bitcoin Liveness, Selfish Mining and Impact of Network Delay

Date 25-Jan-2021

Lecturer: David Tse Scribe: Sicheng Zeng

1 Agenda

- HW 2 out, due next Tues.

2 Liveness

Recall that safety means that if a block is confirmed, then it will stay confirmed. However, safety is not enough if you are not doing anything useful. So, we need another notion that we are making progress and confirming honest blocks.

Definition 2.1. Liveness: A non-zero fraction of honest blocks are confirmed.

Some comments about this definition:

1. We are only considering honest blocks, because non-honest blocks can post junk or be empty.
2. We only require a non-zero fraction, as we have seen before that some blocks could be kicked out of the longest chain by the adversary. As long as a non-zero fraction of honest blocks are confirmed, any honest transaction will eventually be added to a block and to the blockchain.

2.1 Chain Quality

Definition 2.2. Chain Quality (CQ): The long term average fraction of blocks on the longest chain that are honest.

With this new definition, we can redefine liveness as follows:

Definition 2.3. Liveness: $CQ > 0$

We will derive a lower bound on CQ for given λ_h and λ_a and for any attack. As a reminder, λ_h is the hash rate of the honest nodes and λ_a is the hash rate of adversary nodes. To do so, we first introduce a new definition:

Definition 2.4. Chain Growth (g): Long term average growth rate of the longest chain. (blocks/s)

In the worst case, if adversary is able to insert all of its blocks onto the longest chain, the growth rate of adversary blocks on the longest chain would be λ_a.

Since CQ is the ratio of honest blocks to the total number of blocks, and we know that if the growth is g and the adversary can add blocks at a rate of at most λ_a, the chain quality must be lower bounded by

$$CQ \geq \frac{g - \lambda_a}{g}$$

Notice that our answer is in terms of g, which is unknown and can be manipulated by the adversary. Therefore, we can further get the following lower bound on CQ, which does not depend on g:

$$CQ \geq \min_g \frac{g - \lambda_a}{g}$$

By choosing $g = g^* = \lambda_a$, we have $CQ = \frac{g* - \lambda_a}{g*} = \frac{\lambda_a - \lambda_a}{\lambda_a} = 0$. A useless lower bound. But can the adversary make $g = \lambda_a$? Honest nodes should always be following the protocol. Thus, if the adversary is not mining, $g = \lambda_h$. If the adversary is following protocol, then $g = \lambda_h + \lambda_a$. In general, g is greater than or equal to λ_h. From this, we can can get a tighter lower bound:

$$\begin{aligned} CQ &\geq \min_{g \geq \lambda_h} \frac{g - \lambda_a}{g} && (1) \\ &= \min_{g \geq \lambda_h} 1 - \frac{\lambda_a}{g} && (2) \\ &= 1 - \frac{\lambda_a}{\lambda_h} && (3) \end{aligned}$$

Hence,

$$CQ > 0 \text{ if } \lambda_h > \lambda_a$$

2.2 Security Conclusions

Theorem 2.5. *Bitcoin is live if $\lambda_h > \lambda_a$.*

Recall that bitcoin is safe if $\lambda_h > \lambda_a$.

Theorem 2.6. *Bitcoin is secure (i.e. safe & live) if $\lambda_h > \lambda_a$.*

3 Selfish Mining

We can use our chain-quality result 3 to reason not only about the liveness but also about the fairness of Bitcoin.

F2Pool is currently the largest mining pool in bitcoin. Let's say they currently have 20% of the global hash rate. We can imagine F2Pool as an adversary and reason about the situation as follows.

$$\frac{\lambda_a}{\lambda_h} = \frac{20\%}{80\%} = 1/4$$

$$CQ \geq 1 - 1/4 = 0.75$$

Hence, according to our CQ bound, F2Pool can potentially get up to 25% of the block rewards. This might seem unfair, since they only own 20% of the hash rate, and you would expect them to ideally only get 20% of the reward. Using the same formula 3, it can be deduced that if F2Pool has 50% of the hash rate, it could get 100% of the block rewards. Is our bound too loose, and this unfairness actually cannot occur? Or is there an attack that can actually achieve this bound?

As it turns out, the answer is the latter. There is an attack that can achieve this bound, where every adversary block is useful and can replace an honest block on the longest chain. This is called the selfish mining [1].

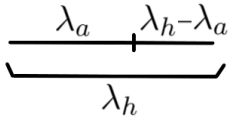

Figure 1: If the lower bound is tight, then every adversary block is useful and can replace an honest block.

Selfish mining attack:

1. The attacker always mines on the block at the greatest level, whether it is private or public. The attacker keeps the block in private and only release it later.
2. When an honest block appears, attacker will release an adversary block at the same level if it has one. (If none, do nothing.)

Following the example from Figure 2, the attacker will mine in private on top of block b_2. Suppose the attacker successfully mines a block, b_3, which it keeps in private. Then, if the honest miners get a block b_4, it will also be a child of b_2, since honest miners are unaware of b_3. At this point, the attacker releases b_3. We assume that the attacker can break ties in its favor. So the next honest block b_5 will be mined on b_3, and the honest block b_4 will be wasted.

Then imagine the attacker gets two private blocks b_6 and b_7. When a honest miner mines b_8, the attacker releases b_6. Then when a honest miner mines b_9, the attacker releases b_7.

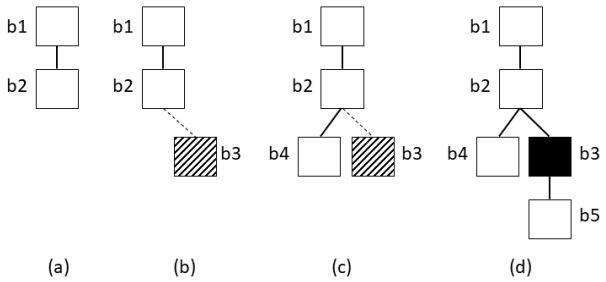

Figure 2: Selfish mining example.

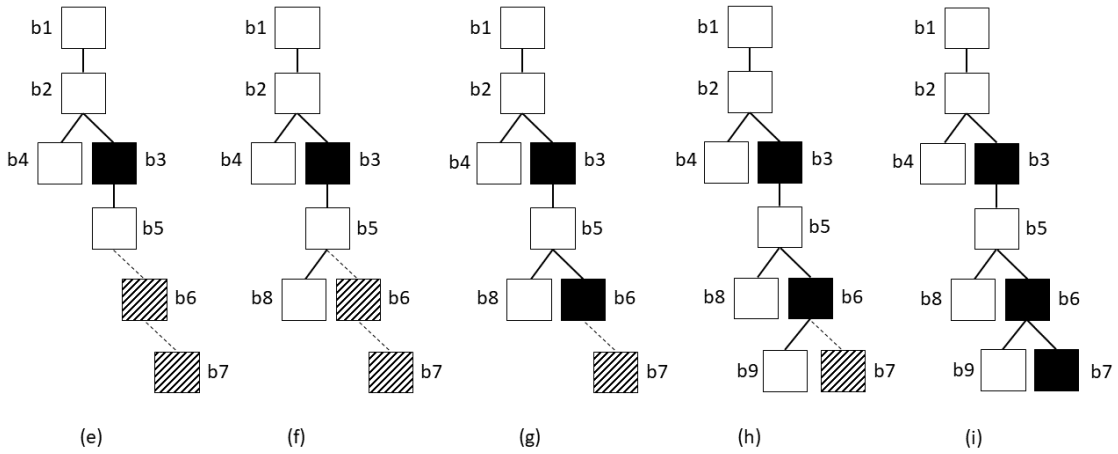

Figure 3: Continuation of selfish mining example.

Every adversary block displaces one honest block. b_3 displaces b_4, b_6 displaces b_8, and b_7 displaces b_9. Therefore, the attack has been successful in matching the lower bound 3.

This is a pretty cool result, as we've shown that the adversary and the honest nodes are not in a symmetrical situation. The attacker has an advantage, because it can mine in private and choose its block release timings. However, even if the attacker can reduce fairness, we still have our security guarantees of liveness and safety whenever the attacker has less hashing power than the honest miners.

4 Network Delay

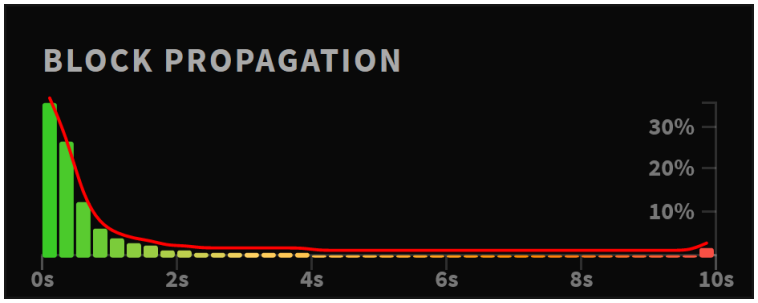

Figure 4: Screencapture from ethstats.net of the block propagation times of Ethereum.

So far, we have been ignoring network delay as a simplification, and because Nakamoto does the same in many of their calculations. However, we will now consider what happens when we add network delay back in, and modify our claims about safety and liveness accordingly.

Block propagation measures the time it takes for a block to get propagated to the worldwide network. This includes processing times from each miner checking the block - proof of work, valid transactions, etc. We can see from Figure 6 that the block propagation time for Ethereum is on the order of a few seconds. The block propagation time of Bitcoin is usually longer.

4.1 Private Attack Analysis

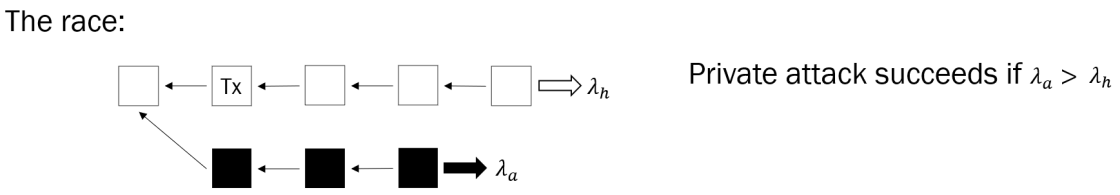

Figure 5: In our original private attack analysis, the private attack succeeds if the attacker (shown in black) mines k blocks before the honest nodes mine k blocks, and this happens if $\lambda_a > \lambda_h$.

Now that we are considering network delay, we must modify our safety analysis. In our previous model of Bitcoin with no network delay, the only way to have forking is from an adversarial attack. But now, with network delays, there is a possibility to have **natural forking**, a.k.a. forking among honest blocks. After a block is published, it's possible that a different honest miner that has not seen the published block publishes a new block on the same level, so that two honest blocks are published on the same parent.

Since a block is published every 10 minutes, natural forking is unlikely to occur in Bitcoin, but with faster block publishing times like in Ethereum, where the block time is around 14 seconds, forking happens more often. (Ethereum has natural forking in 10-15% of the levels.)

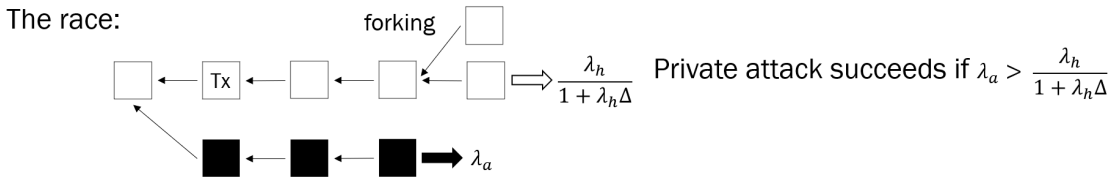

Figure 6: With consideration of network delay, natural forking can occur among honest blocks, so the hashing rate of honest miners is decreased, and the private attack succeeds if $\lambda_a > \frac{\lambda_h}{1+\lambda_h\Delta}$.

Because of network delay, some honest miners will be mining on older blocks and their work will be wasted, and therefore g, the chain growth is reduced. Let Δ be the network delay. Then, the parameter

$$\lambda_h\Delta =$$

is the expected number of honest blocks mined during the network delay, and can be taken as a measure of how much forking there is in the network. The larger this number, the more forking there is. In Q. 3 of Homework 2, you will show that, under the assumption that there are large number of honest miners each with infinitesimal mining power, the chain growth rate of the honest chain is reduced from λ_h to

$$\frac{\lambda_h}{1+\lambda_h\Delta}.$$

Note that attacker blocks could all be mined in a centralized chain, so it's possible that there is no forking in the adversary chain, growing still at rate λ_a.

With our adjustments, a private attack succeeds if

$$\lambda_a > \frac{\lambda_h}{1+\lambda_h\Delta}$$

References

[1] I. Eyal and E. G. Sirer. Majority is not enough: bitcoin mining is vulnerable. *Commun. ACM*, 61(7):95–102, 2018.