ECE 595: Foundations of Blockchain Systems **October 12, 2020**

Lecture 4

Lecturer: Sreeram Kannan *Scribe: Wenjie Nan*

The lecture starts with discussing the consensus & persistence in longest chain. Then it digs into the structure of Bitcoin system–**header chain**, including the advantages of header chain, the concept of commitment and the transaction grinding. Then some basic information about the Bitcoin system is introduced, including the website of inspecting the Bitcoin system and some intrinsic parameters in Bitcoin system. At last, the economics of mining will be discussed, especially the strategy – **mining pools**, including the reason for using such strategy and how mining pools work without trusting on each other.

4.1 Consensus & persistence.

The nodes in blockchain system which has a decentralized structure do not need to always have uniform recognition of the longest chain. The longest chain can vary on different nodes at the same time, or same nodes at different time since there may be some branches of blockchain. To achieve the consensus & persistence, two requirements should be satisfied,

- Match among different nodes at same time. (Consensus)
- Match among different times of same node. (Persistence)

Match is defined as the similar blocks in the longest chains from different perspectives. Furthermore, to show the persistence along the time, it is necessary to show that the longest chain will not change or reverse significantly with time. Such persistence require the majority of computational power is controlled by the honest nodes.

4.2 Deeper into Bitcion system.

The blockchain structure will be discussed, especially how different blocks connect to each other. As mentioned in previous lectures, the miners can choose to create a hash of the block that they want to connect to (the previous block). The previous block will be denoted as B_0 where the hash of block is $H(B_0)$. Then they can hash the previous hash $H(B_0)$ and random nonces to find the nonce that

$$H(H(B_0), Nonces) < Threshold.$$

After a miner luckily find a nonce that satisfies the inequality above before others, a new block connecting to B_0 is created. Since all nodes in the blockchain system can easily check the inequality, then they will know that the new block is pointed to B_0.

The method above is obviously executable to create a series of blocks, however, another method has some difference with the previous one is used in Bitcoin system – **Header chain**. The header chain means that instead of hashing the whole block, now the miners only hash the header of block they want to connect. The header of the block B_1 can be denoted as $\alpha(B_1)$. $\alpha(B_1)$ contains several different information,

- The hash of the previous block's (B_0) header : $H(\alpha(B_0))$.
- The commitment of all transactions in B_1 : $CMT(Tx_{B_1})$.

- Some nonce that make inequality $H(H(\alpha(B_0)), CMT(B_1), Nonce) < Threshold$ hold.

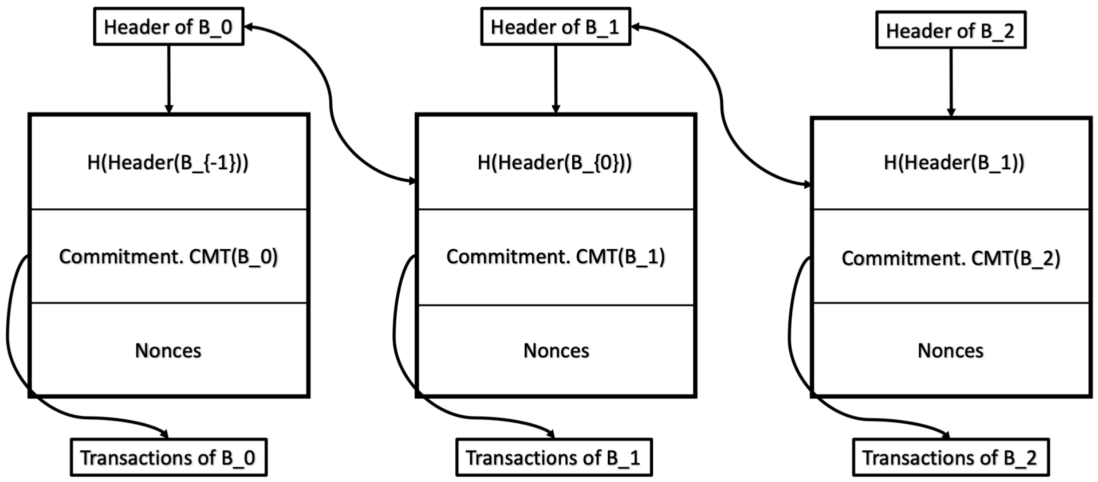


Figure 4.1: Header chain

4.2.1 Advantages of header chain.

The header chain structure can help nodes to identify the longest chain without downloading all blocks and checking the hash function of these blocks. For instance, a new node will receive the information from different nodes connecting to it. Different nodes may have different perspectives of longest chain, the new node only need to download the headers of blocks from different nodes to identify the longest chain. After recognizing the longest chain, the new node only need to download the blocks of the longest chain.
Otherwise, the new nodes may need to download all different blocks from different nodes. Some nodes may send some juke, invalid blocks to new nodes. As a result, the header chain structure also protect the new nodes from such attack.

4.2.2 Commitment.

The definition of ”commit” is that, a string S is saying to commit on a data item D_1 means that given the string, it is practically impossible to find a different data item D_2 which has similar string S. An example is the string S created from the hash function. Since it is practically impossible to find D_2 where $H(D_1) = H(D_2)$, then $S = H(D_1)$ commits to data item D_1. Then commitment is defined as a string that commits on a data item.
The commitment on the all transactions in B_1 means that the other nodes can always check the transactions in block B_1 that if the commitment of these transactions equal to the commitment in the header. If someone modify the transactions, it is impossible for them to find similar commitment using different data item. Then the only way they will be able to modify the transactions is to change the whole block. In previous lecture, such attack has also been discussed that attackers need to create a longer chain from the block they want to modify.
The commitment in real Bitcoin system is calculated based on hash function, but it is more complex than hash function. It will calculate hash value for each transaction then use Merkle Tree to get the commitment. [2]

4.2.3 Transaction grinding.

Since the miners can always choose how many transactions to make the commitment, now it seems that there are 2 degree of freedom of calculating the hash function. One degree is that miners can choose a random subset of transactions in his block, another degree is the random nonces. However, the miners will never choose to use a random subset to calculate the hash function. There are 2 reasons.
First reason is that no matter the miners randomly choose the subsets of transactions or nonces, the only thing that help miners to create more blocks is to use more computational power. The property of the hash function determines that changing nonces or subsets are similar for calculate hash.
Second reason appears since the first reason is not convincing enough. Why do miner never use subsets of transactions in his block if changing subsets and nonces are similar? The reason is that the if the miners want to maximize his profits, he should include all transactions into the commitment. Then these transactions will be valid since the previous discussion in Commitment shows that others can check if the transactions match the commitments. Then miners can make money from these valid transactions.

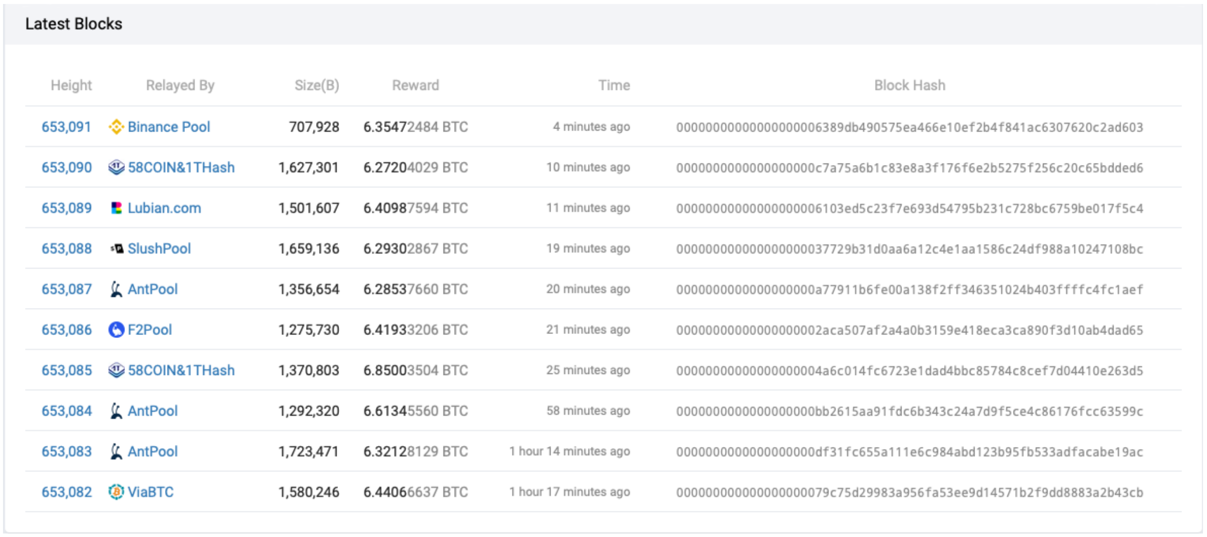
4.3 Bitcoin system basics and related calculation.

4.3.1 Bitcoin system basics.

he website `http://btc.com` offers real time information about the Bitcoin system. The information like the depth of Bitcoin chain, mining companies, hash value of each block and so on, can be found on the website.
The mining process is how new blocks in Bitcoin system are created, and the miners need to show their computational power called proof of work. With more or less people are mining, the total computational power will increase or decrease. To keep the constant speed of generating new blocks, which is average 10 minutes 1 block, the difficulty of mining will be adjusted every 2016 blocks. Both 10 minutes and 2016 blocks are fixed parameter inside the Bitcoin system.
The difficulty τ can be defined as the threshold of the hash value that $H < \tau$, H is defined at section 4.2 as $H = H(H(\alpha(B_0)), CMT(B_1), Nonce)$.
The figure 1 shows the hash values at 2020-10-16 contains 19 leading zeros. The number of leading zeros of hash value, denoted as l, can be thought as an estimation of the difficulty τ. Because larger l means more leading zeros in hash value, then the threshold will be smaller. It means that it is more difficult to find $H < \tau$. Given the fixed total computational power, if l is too large, the threshold will be too small, the difficulty will enlarge the time interval between blocks; if l is too small, too many blocks will be created in very short time period. The hash values on the website is in hexadecimal. As a result, $l = 76$ in binary system. Then the difficulty $\tau \sim 2^{256-l}$. Then $H < 2^{256-l}$.
The figure 4.2 also shows that the block reward as present is almost 6.25 BTC, and it is 12.5 BTC a year age. Reason is the bitcoin system will half the reward after n blocks are created to keep the total number of BTC is 21 million. $80 - 90$ percents of BTC have already be mined so far. [1]

4.3.2 Related calculation

Example 1: The average number of times of calculating hash value to create a new block at 2020-10-16. Using the information that the hash values contain 19 leading zeros in hexadecimal. The



Latest Blocks

Height	Relayed By	Size(B)	Reward	Time	Block Hash
653,091	Binance Pool	707,928	6.35472484 BTC	4 minutes ago	00000000000000000006389db490575ea466e10ef2b4f841ac6307620c2ad603
653,090	58COIN&1THash	1,627,301	6.27204029 BTC	10 minutes ago	0000000000000000000c7a75a6b1c83e8a3f176f6e2b5275f256c20c65bdded6
653,089	Lubian.com	1,501,607	6.40987594 BTC	11 minutes ago	00000000000000000006103ed5c23f7e693d54795b231c728bc6759be017f5c4
653,088	SlushPool	1,659,136	6.29302867 BTC	19 minutes ago	000000000000000000037729b31d0aa6a12c4e1aa1586c24df988a10247108bc
653,087	AntPool	1,356,654	6.28537660 BTC	20 minutes ago	0000000000000000000a77911b6fe00a138f2ff346351024b403ffffc4fc1aef
653,086	F2Pool	1,275,730	6.41933206 BTC	21 minutes ago	00000000000000000002aca507af2a4a0b3159e418eca3ca890f3d10ab4dad65
653,085	58COIN&1THash	1,370,803	6.85003504 BTC	25 minutes ago	00000000000000000004a6c014fc6723e1dad4bbc85784c8cef7d04410e263d5
653,084	AntPool	1,292,320	6.61345560 BTC	58 minutes ago	0000000000000000000bb2615aa91fdc6b343c24a7d9f5ce4c86176fcc63599c
653,083	AntPool	1,723,471	6.32128129 BTC	1 hour 14 minutes ago	0000000000000000000df31fc655a111e6c984abd123b95fb533adfacabe19ac
653,082	ViaBTC	1,580,246	6.44066637 BTC	1 hour 17 minutes ago	000000000000000000079c75d29983a956fa53ee9d14571b2f9dd8883a2b43cb

Figure 4.2: Screen shot of `http://btc.com` at 2020-10-16

probability of one calculation to get the object hash value will be

$$P(H < 2^{256-76}) = \frac{2^{180}}{2^{256}} = 2^{-76}.$$

Then averagely, 2^{76} calculations are needed to mine a new block.
Example 2: The average number of times of calculating hash value per second. Recalling that the Bitcoin system is designed to generate a new block per 10 minute. Then hashrate will be

$$\frac{2^{76}}{600} = 1.25 \times 10^{20} = 125EH/s \qquad (1EH/S = 10^{18} times/s).$$

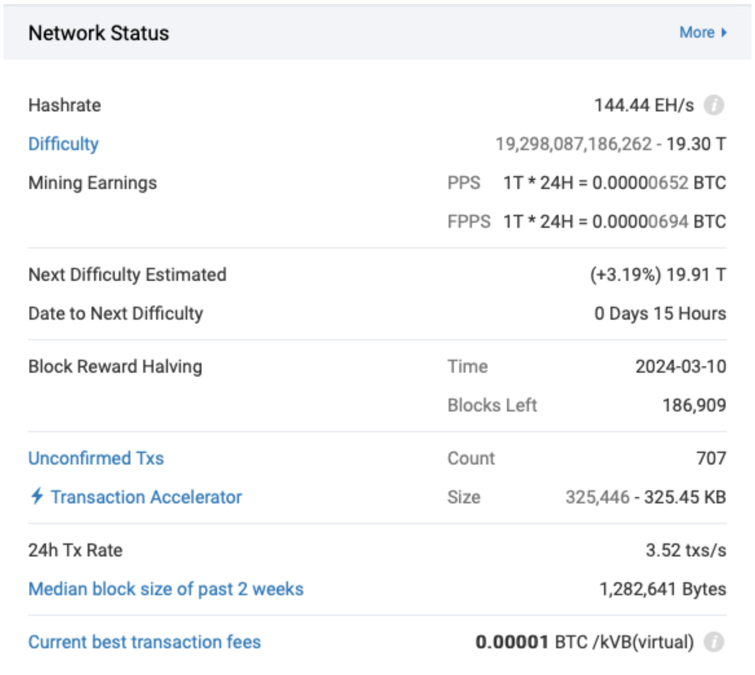
As showed in Figure 4.3, the hashrate on the website is close to the estimation.

4.4 The economics of mining – mining pools.

There are two reasons for mining.

- Block reward after mining a new block.
- Transaction fee on every transaction.

Every transaction included in the block will pay the transaction fees to the miner who created the block. The transaction fees are not prescribed in the system, it is based on market. If some transaction fees are low, the miners can choose not to include these transactions. Currently, the block reward is much greater than transaction fees. The block reward now is 6.25 BTC per block. To get the reward, miners need to calculate $2^{76} \sim 7 \times 10^{22}$ hash values. And the price for cloud miners will be $0.6\$/(10^{10}H/s)$. Obviously, it is non-profitable for cloud miners.
To make profit, some strategic behaviors are observed. For instance, mining pools. The most interesting part of mining pools is that this kind of cooperative mining does not rely on trust.



Network Status		More ▸
Hashrate		144.44 EH/s
Difficulty		19,298,087,186,262 - 19.30 T
Mining Earnings	PPS	1T * 24H = 0.00000652 BTC
	FPPS	1T * 24H = 0.00000694 BTC
Next Difficulty Estimated		(+3.19%) 19.91 T
Date to Next Difficulty		0 Days 15 Hours
Block Reward Halving	Time	2024-03-10
	Blocks Left	186,909
Unconfirmed Txs	Count	707
Transaction Accelerator	Size	325,446 - 325.45 KB
24h Tx Rate		3.52 txs/s
Median block size of past 2 weeks		1,282,641 Bytes
Current best transaction fees		**0.00001** BTC /kVB(virtual)

Figure 4.3: Screen shot of `http://btc.com` at 2020-10-16

4.4.1 The reason for mining pools.

For a miner A, if computational power of A is fixed and the total computational power of all miners is also fixed, then the expected return of A is fixed no matter A mines alone or cooperates with others. However, if A mines alone, the variance of return will be very large since there are so many miners. Furthermore, since the total number of blocks that can be generated in a period is almost a constant, most individual miners can be expected have return 0 after mining sometime. However, if many miners cooperate together that once one of the miners gets lucky, then all miners can make profit based on their computational power. As a result, mining pools reduce the variance of return while keeping the expected return same. Miners who do not want to take too much risks will join the mining pools.

4.4.2 The mechanism for mining pools without trust.

The previous discussion seems to include some mechanism of splitting the profit based on the trust on each other or trust on some kind of centralized coordinator. However, the special properties of Bitcoin chain can make miners work together without trusting on anyone else.
The first property is that miners have no power to modify the blocks after mining even the blocks are created by them. It means that if a block is created, broadcasted and included in the longest chain, the information on the block will be always be accepted unless an even longer chain is created which does not include the block. The property determines that the miners can make some agreement before mining, for instance, they can write the profit into the block before mining. Once the block is mining successfully, then the profit will be distributed according to the pre-decided way inside the block.
However, the first property can not make sure that all miners will work as the way they determined, for instance, the miners can mine the block just for themselves after joining the mining

pools. Then the second property of block chain is that everyone can check the hash value of other miners in mining pool to make sure that they are working on the same block. Since it is practically impossible to generate same hash value for different input, there is no way for miners to cheat other miners without being found. As a result, the miners in the mining pool only need to check the hash values from others every some time period t, for instance, every second.
Additionally, the miners also need to confirm that others are using the computational power that they claim they have. The computational power can be checked by hash value, too. Although for some miners, it is impossible to get value $H < \tau$, but they will definitely generate some hash values with some leading zeros. The leading zeros is their partial proof of work. They need to send the partial proof of work every some time period t to others.
The one last problem is that some miners may cheat after some time T. It seems that other miners will waste T, actually other miners only waste time t which is the period of sending the partial proof of work. The reason is the expected returns from $0 \rightarrow T - t$ is still same as the returns written into the block they are mining. The expected return decrease only in $T - t \rightarrow T$ because someone is cheating. Then other miners can always stop working to kick that one out.

References

[1] Investopedia website. *Bitcoin mining*.

[2] SelfKey website. *What is a Merkle Tree and How Does it Affect Blockchain Technology?*