ECE 595: Foundations of Blockchain Systems **November 9, 2020**

Lecture 12: Throughput

Lecturer: Sreeram Kannan *Scribe: Zichen Liu*

Outline: In this lecture, we first review the concept of throughput and explain the scalability difficulty of Nakamoto consensus protocol. Then we introduce some protocols that dedicate to achieve both high throughput and full security, including **Bitcoin-NG**, **Inclusive**, **GHOST** and **Prism1.0**.

12.1 Bitcoin's Throughput

In the previous lectures, we have already discussed the throughput, which is one of the basic concepts in blockchains, reflects the speed at which a blockchain processes transactions. In this section, we will firstly revisit the throughput of Bitcoin system, and then further discuss the limitations of throughput and the difficulty of improving throughput under Bitcoin system.

12.1.1 Review Throughput

- **Notation**

β	adversarial ratio	$[\beta] = 1$
β^*	maximum adversarial ratio	$[\beta^*] = 1$
λ	block rate	$[\lambda] = (num\, of\, blocks)/second$
λ_a	adversarial block rate	$[\lambda_a] = (num\, of\, blocks)/second$
λ_h	honest block rate	$[\lambda_h] = (num\, of\, blocks)/second$
B	block size	$[\mathrm{B}] = (num\, of\, transactions)/block$
C	network capacity	$[\mathrm{C}] = (num\, of\, transactions)/second$
D	propagation delay	$[\mathrm{D}] = second$
R	throughput	$[\mathrm{R}] = (num\, of\, transactions)/second$
Δ	block delay	$[\Delta] = seconds/block$

The adversarial ratio is defined as $\beta := \frac{\lambda_a}{\lambda_a+\lambda_h} = \frac{\lambda_a}{\lambda}$, represents the fraction of adversarial mining power to the total mining power. The λ here represents the total mining power, and we have

$\lambda = \lambda_a + \lambda_h$. The lower bound of the longest chain growth rate is $\frac{\lambda_h}{1+\lambda_h\Delta}$. Let's assume each block contains same number of transactions, i.e. all blocks have the same block size, which is B. Based on the notations listed above, the throughput (R) can be computed by,

$$\mathrm{R} = \frac{\lambda_h}{1+\lambda_h\Delta} \times \mathrm{B} \tag{12.1}$$

In addition, based on the network model we use, the block latency Δ contains two parts, namely propagation delay (D) and download delay. The block latency Δ is expressed as,

$$\Delta = \mathrm{D} + \frac{\mathrm{B}}{\mathrm{C}}$$

where the second term $\frac{\mathrm{B}}{\mathrm{C}}$ is download delay. It can be understood as the download time per block, so the unit of dewnload delay is seconds per block. Note that the expression of the block latency is just from the network model, it may not fully express the situation in the real world. For example, we can also take the queuing delay, which is in stochastic form, into consideration. The queuing delay takes the potential network congestion into account, and assume that the blocks or transactions will not be broadcast immediately when they are mined or signed. The network model also contains the bandwidth (capacity) constrain, which can be expressed as,

$$\lambda\mathrm{B} \leq \mathrm{C} \tag{12.2}$$

Considering the security requirement of longest chain protocols — the growth rate of honest chain should be larger than that of the adversarial chain — we have to choose λ_a and λ_h to satisfy the following inequality

$$\frac{\lambda_h}{1+\lambda_h\Delta} > \lambda_a \tag{12.3}$$

By substituting $\frac{\lambda_a}{\lambda_a+\lambda_h}$ with β, (12.3) can be translated to

$$\frac{1-\beta}{\beta} > 1 + \lambda_h\Delta \tag{12.4}$$

From (12.4), we have

$$\beta < \frac{1}{2+\lambda_h\Delta}$$

If we want $\beta \to \frac{1}{2}$, we need $\lambda_h\Delta \to 0$.

Hence, we define the maximum adversarial ratio β^* as the maximum possible value of β. By substituting Δ with $\mathrm{D} + \frac{\mathrm{B}}{\mathrm{C}}$, we can get

$$\beta^* = \frac{1}{2+\lambda_h(\mathrm{D}+\frac{\mathrm{B}}{\mathrm{C}})} \tag{12.5}$$

We can view the throughput as the ratio of it to the network bandwidth by transforming the (12.1) into the proportion form, which can be written as

$$\frac{\mathrm{R}}{\mathrm{C}} = \frac{\lambda_h \frac{\mathrm{B}}{\mathrm{C}}}{1 + \lambda_h(\mathrm{D} + \frac{\mathrm{B}}{\mathrm{C}})} \tag{12.6}$$

By replacing the λ in (12.2) with λ_h, (12.2) can be written as

$$\frac{\lambda_h}{1-\beta}\mathrm{B} \leq \mathrm{C} \tag{12.7}$$

12.1.2 Ideal Properties

(12.5), (12.6) and (12.7) are the core equations we will be using in the subsequent parts. In these three equations, the propagation latency D and the network bandwidth C are given by the objective system, and are unchangeable, so the only two variables that we can tune are λ_h and B. Our goal is to make both the throughput and the maximum adversarial ratio to approach their theoretical limits, that is, the throughput $\beta^* \to 1/2$ and the throughput to approach the network bandwidth, i.e. $\mathrm{R} \to \mathrm{C}$, respectively.

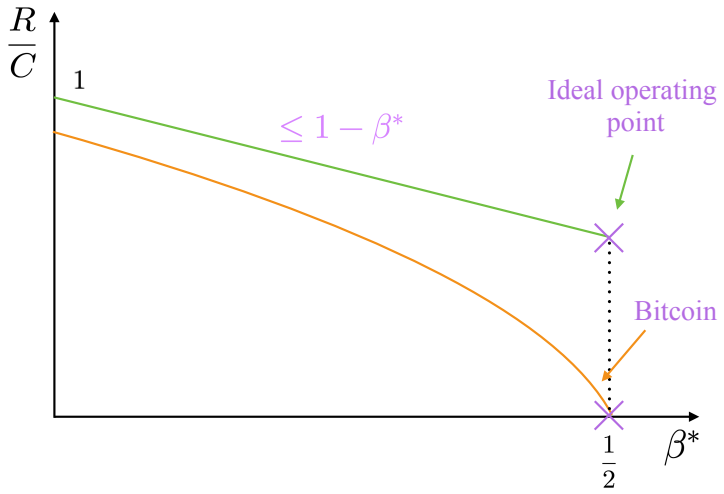

Figure 12.1: Throughput bandwidth ratio vs. Maximum adversarial ratio β^*

Fig.12.1 shows how the upper bound of bandwidth utilization $\frac{\mathrm{R}}{\mathrm{C}}$ changes with different maximum adversarial ratio β^*. The green curve in Fig.12.1 is the ideal curve , while the yellow curve is the upper bound that Bitcoin can achieve. Note, the bandwidth utilization of the ideal curve is not 1, since we must assume the worst case, that is, all the adversarial blocks will be wasted

block and won't contain any legal transactions. In this case, β^* can be understood as the ratio of bandwidth wasted by adversaries, the ideal bandwidth utilization curve is $1-\beta^*$.

12.1.3 Difficulty of Bitcoin

There is a trade-off between throughput and security in Nakamoto longest chain protocol and that is the difficulty that Bitcoin system can not overcome. In this subsection, we will explain why Bitcoin system performs like the yellow curve in Fig.12.1, by showing several examples.

1. **E.g.1**: Suppose we want $\beta^* \to 1/2$, according to (12.5), we need

   $$\lambda_h(\mathrm{D}+\frac{\mathrm{B}}{\mathrm{C}}) \to 0$$

   which implies

   $$\begin{cases} \lambda_h \mathrm{D} \to 0 \\ \lambda_h \frac{\mathrm{B}}{\mathrm{C}} \to 0 \end{cases}$$

   According to (12.6), we have $\frac{\mathrm{R}}{\mathrm{C}} < \frac{\lambda_h \mathrm{B}}{\mathrm{C}} \longmapsto \frac{\mathrm{R}}{\mathrm{C}} \to 0$, which goes against our intention.

2. **E.g.2**: Suppose we fix B and tune λ_h. When λ_h satisfies $\lambda_h(\mathrm{D}+\frac{\mathrm{B}}{\mathrm{C}}) = \frac{1}{100}$, we will get

   $$\beta^* = \frac{1}{2+\lambda_h(\mathrm{D}+\frac{\mathrm{B}}{\mathrm{C}})} = \frac{1}{2+0.01} \to \frac{1}{2}$$

   However, under this circumstance, the bandwidth utilization ratio is

   $$\frac{\mathrm{R}}{\mathrm{C}} = \frac{\lambda_h \frac{\mathrm{B}}{\mathrm{C}}}{1+0.01} \to \frac{\lambda_h \mathrm{B}}{\mathrm{C}} = \frac{1}{100(\mathrm{D}+\frac{\mathrm{B}}{\mathrm{C}})} \times \frac{\mathrm{B}}{\mathrm{C}} \leq \frac{1}{100}$$

   which also goes against our intention.

The above two examples show that when we require the Bitcoin system to be secure ($\beta^* \to 1/2$), the throughput will inevitably drop to a very low level, which is far from the ideal point. In Bitcoin, we can only increase throughput by increasing mining rate λ_h to make more blocks in the same amount of time and block size B to pack more transactions into each block, but both of them will lead to forking problem, thereby hurting the blcokchain's security.

12.2 Other high throughput protocols

So far we discussed the thoughput of Bitcoin and the difficulty Bitcoin faces. Because the above-mentioned difficulty in Bitcoin, many protocols that claim to be able to achieve high throughput

while ensuring full security have been proposed. In this section, we will mainly go through several high-throughput protocols, including Bitcoin-NG, Inclusive, GHOST and Prism1.0.

12.2.1 Bitcoin-NG

One of the earliest protocols to solve the poor throughput of Bitcoin is Bitcoin-NG (Next Generation) [1] by Eyal and Renesse in 2016. The basic idea of Bitcoin-NG is decoupling Bitcoin's blockchain operation into two planes: *leader election* and *transaction serialization* by dividing the block into two types, namely *key block* and *microblock*. In addition to this, Bitcoin-NG also follows the single longest chain structure and shares the same trust model as Bitcoin. The structure of Bitcoin-NG blockchain is shown in Fig.12.2.

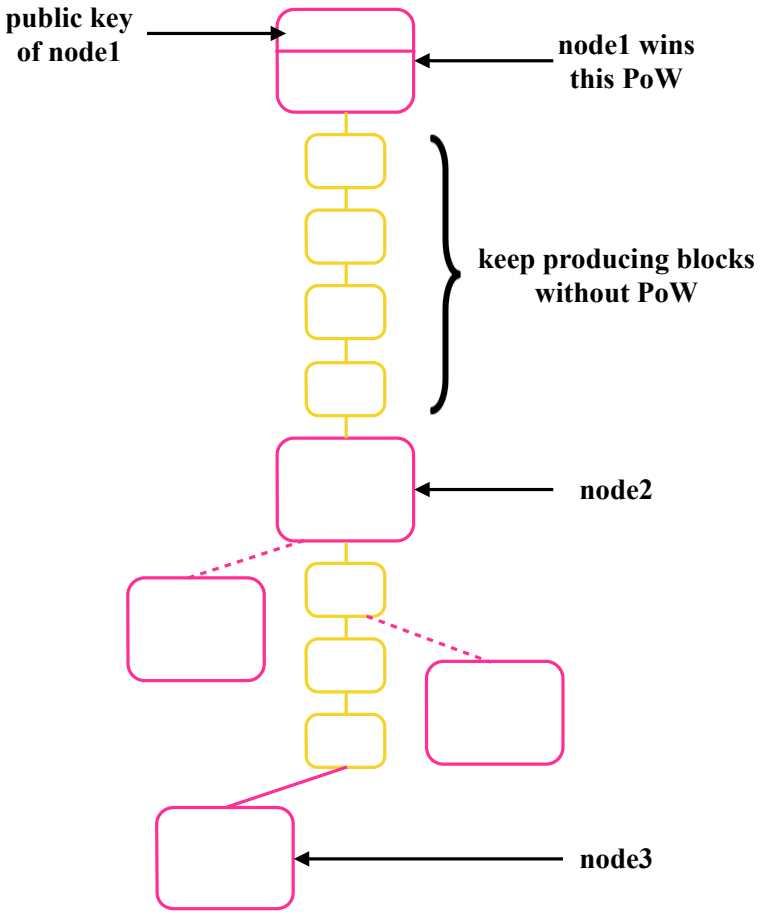

Figure 12.2: Bitcoin-NG structure

The pink blocks in Fig.12.2 are key blocks and the yellow blocks are microblocks. Same as Bitcoin, a node is elected the leader by winning the *Proof of Work* (PoW) among key blocks. They perform as leaders of their successor microblocks. Once a leader is chosen, it is entitled to generate microblocks unilaterally until a new leader is chosen, marking the end of the former's epoch. Microblocks are used to pack transactions and don't fork, since they are signed by their respective leader nodes, instead of being mined by PoW. Note that microblocks have no weight, so that the length of the chain only depends on the depth of key blocks in the chain and microblocks don't contribute to Bitcoin-NG's security.

Suppose in Fig.12.2, if node1 wins the PoW puzzle and adds its key block to the blockchain, the following microblocks will be signed by node1's private key periodically until node2 is elected as the next leader. Note, node1 needs to add its public key to the header of its key block for microblock verification. While microblocks being released periodically, other miners will continually try to mine the next key block based on the newest block. A key block contains the reference to the previous block (either a key block or a microblock, usually the latter, since the production rate of microblocks is much larger than that of key blocks). As shown in Fig.12.2, after the second key block mined by node2 being attached to the main chain, miners will firstly try to mine on this key block. If no legal key block is broadcast before the next microblock is released, miners will mine on the new microblock. The above process will repeat until the new leader node is elected. Although no forking will occur in a series of microblocks, forking does occur sometimes at key blocks. In case of a fork, the nodes pick the branch with the most weight, aggregated over all key blocks.

By introducing two new parameters, namely inter-arrival time and size of microblocks, Bitcoin-NG decouples its security and throughput. Specifically,

- Adjust the key blocks' parameters for security
- Adjust the microblocks' parameters for high throughput

Given the above properties, we can maximize the parameters of microblocks to fully fill the bandwidth, while tuning the parameters of key blocks to achieve full security. Although the benefits of Bitcoin-NG are obvious, it also has two shortcomings,

1. The confirmation latency doesn't improve compared to Bitcoin. Because the confirmation of transactions in Bitcoin-NG has k-deep requirement on key blocks;

2. Not fully decentralized. This is because every key block is in charge of its successor mi-

croblocks. Keyblocks can easily implement double spending attacks.

12.2.2 Inclusive

Inclusive [2] was proposed by Sompolinsky and Zohar in 2015. Compared with Bitcoin, Inclusive's solution to the waste caused by forking is to introduce a special link, called *reference link*. By introducing the reference link, Inclusive is the first blockchain protocol to introduce directed acyclic graphs (DAG).

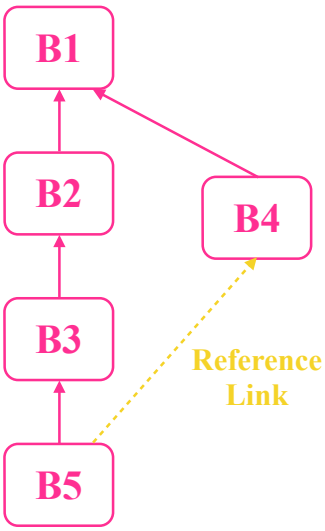

Figure 12.3: Inclusive structure

As illustrated in Fig.12.3, when B5 notices that B4 has not been included, a reference link from B5 to B4, which is drawn as a dotted line, will be constructed. The function of a reference link is same as write all the transactions from the referred block into the referring block. Since no forking waste and all the honest blocks will eventually be added, we have

$$\frac{\mathrm{R}}{\mathrm{C}} = \frac{\lambda_h \mathrm{B}}{\mathrm{C}} \tag{12.8}$$

However, Inclusive also obeys (12.5) and (12.7). Combining (12.5), (12.7) and (12.8) together, if we want $\beta^* \to 1/2$, we still need $\frac{\lambda_h \mathrm{B}}{\mathrm{C}} \to 0$, i.e. $\frac{\mathrm{R}}{\mathrm{C}} \to 0$. In conclusion, Inclusive doesn't waste blocks, thus allows higher throughput compared with Bitcoin, but the improvement is limited.

12.2.3 GHOST

GHOST [3] was also proposed by the Inclusive team in 2015. GHOST has been introduced in detail in our previous lectures. It is vulnerable to balance attack and its throughput is proven to be lower than Bitcoin.

12.2.4 Prism 1.0

Prism 1.0's fundamental idea is similar to Bitcoin-NG, which is to divide blocks that play multiple roles in Bitcoin into multiple types of functionally specific blocks. In Prism 1.0, two types of blocks are defined. As illustrated in Fig.12.4, the yellow blocks on the left side are called transaction blocks. Transaction blocks are only used to store transactions, and don't have any chain structures between them. By solving relatively easy cryptographic puzzles, miners can mine transaction blocks. The pink blocks on the right side are called *proposer blocks*. Each proposer block contains contains hash values of the transaction blocks that it referred. Proposer blocks are small — since they only contain reference links toward transaction blocks — and thus can be propagated fast without significant impact on latency.

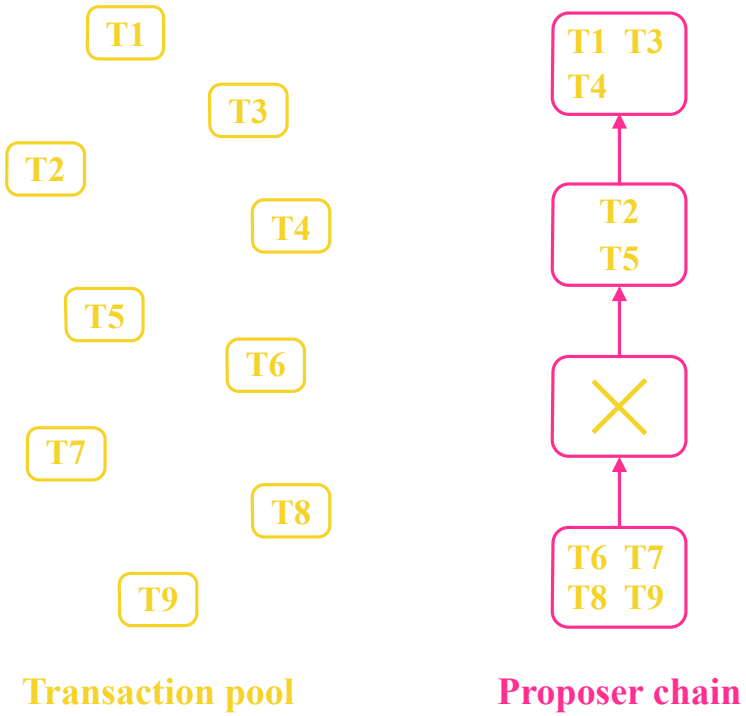

Figure 12.4: Prism 1.0 structure

Let's denote the size of proposer block as $\mathrm{B_p}$ and the size of transaction block as $\mathrm{B_t}$. We can make the block rate of proposer to be small, so that the proposer chain is not easy to generate forking. Therefore, by adjusting the relative threshold of mining transaction block and proposer block, we can increase throughput while maintaining full security.

In addition, in Prism1.0, if a malicious miner intends to waste blocks and doesn't add any transactions into the proposer block, as the third proposer block in Fig.12.4 shows. In this case, other miners will also add transaction blocks that they have received but have not yet been referred. That is, every transaction block will eventually be added.

Precisely, in Prism1.0, miners will only consider the transaction blocks be propagated before the current time minus one timeslot. The timeslot here represents the length of delay in the network, or in other words, every transaction block generated with $timestamp < current\,time - timeslot$ will be received by all of the miners. In this case, the proposer block containing this kind of transaction blocks will be widely accepted after it is produced.

The features of Prism1.0:

1. With high throughput and full security, you can adjust the parameters to make the $\frac{\mathrm{R}}{\mathrm{C}} - \beta^*$ curve of Prism1.0 as close to the ideal curve as possible;

2. Every honest transaction block eventually gets referred. Therefore, determining mining rewards by transaction block yields fair rewards. On the contrary, in Bitcoin, if the mined block doesn't belongs to the main chain, the miner will not be rewarded.

This reward method is called *FruitChain*[4]. FruitChain is a reward distribution mechanism for incentive optimal instead of bandwidth efficiency, based on transaction block. In addition to fair rewards, this reward method also reduces the variance of the rewards. This is because mining the transaction block is much more easier than mining proposer block, hence more participants will be rewarded and thereby reducing the rewards variance. This kind of design makes the existence of the mining pool no longer necessary. But Prism1.0 cannot improve confirmation latency. Because it still needs to meet the k-deep requirements of proposer chain, it will consume the same time as Bitcoin to confirm a transaction.

References

[1] Ittay Eyal, Adem Efe Gencer, Emin Gün Sirer, and Robbert Van Renesse. Bitcoinng: A scalable blockchain protocol. In NSDI, pages 45–59, 2016.

[2] Yoad Lewenberg, Yonatan Sompolinsky, and Aviv Zohar. Inclusive block chain protocols. In International Conference on Financial Cryptography and Data Security, pages 528–547. Springer, 2015.

[3] Yonatan Sompolinsky and Aviv Zohar. Secure high-rate transaction processing in bitcoin. In International Conference on Financial Cryptography and Data Security, pages 507–527. Springer, 2015.

[4] R. Pass and E. Shi. Fruitchains: A fair blockchain. In Proceedings of the ACM Symposium on Principles of Distributed Computing. ACM, 2017.