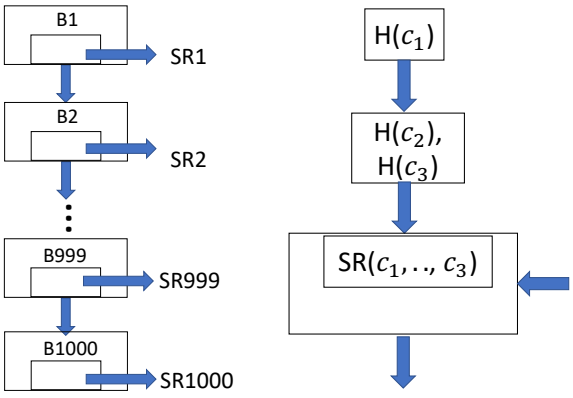
ECE 595: Foundations of Blockchain Systems **November 30, 2020**

Lecture 19

Lecturer: Sreeram Kannan *Scribe: Lyutianyang Zhang*

19.1 Difference between sharded and non-sharded Statement Commitment

For state commitment in Ethereurm (non-sharded), each block is corresponding to a single state root. Therefore, it is very easy for a new block to join the blockchain (bootstrapping), i.e., one can simply chain the state root with the latest published block, as shown in Fig. 1(a). However, in the sharded case, The main chain structure is shown in Fig.1(b), in which hashes of c_1, c_2, and c_3 are calculated and chained together and are chained to the state root together, i.e., $SR(c1, c2, c3)$. The corresponding state commitment is constructed as in Fig.19.2, where accounts and balance represents the state, Jiarong, Hao, and Liu owns 100, 200, and 10 dollars respectively. Hash 1 of Jiarong and Hao's rows are calculated and hash 2 of Liu's row and its previous row is also calculated. Then the hash of hash 1 and 2 are calculated as the Merkle root-state root, which is used for quick verification of the transactions.



(a) Non-Sharded Block Chain (b) Main Chain: Sharded Block

Figure 19.1

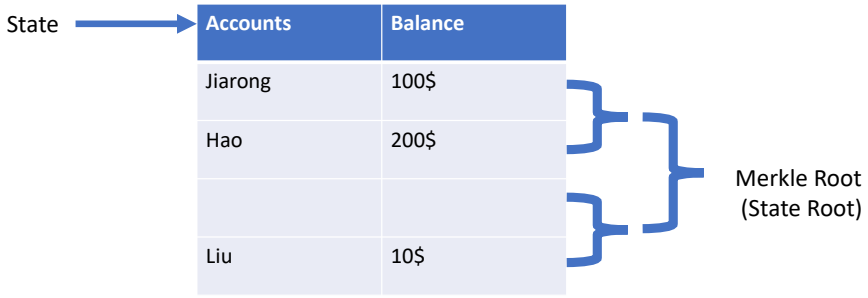


Figure 19.2: State Commitment: Sharded Block

19.2 Key Idea: Interactive State Commitment

In the interactive state commitment, node U mines blocks and posts $SR(1) = SR(B_1)$ to $SR(5) = SR(B_1, B_2, B_3, B_4, B_5)$, sequentially where B_1, ..., B_5 are 5 blocks. For node X, it senses some incorrectness based on its own information and it challenges $SR(3)$ and responds to node u by zooming in to the first $SR(i)$ that was wrong, i.e., $SR(3)$. Then, Node U takes calculates intermediate states roots inside the block B_3 abd posts it. The calculation step is shown in Fig.19.3. Then,

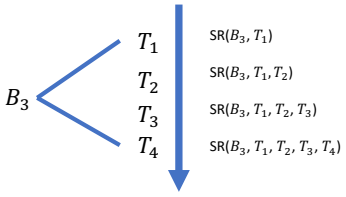


Figure 19.3: Calculate Intermediate State Roots.

for example, Node X disagrees with the $SR(B_3.T_1, T_2)$, although it agrees with the $SR(B_3, T_1)$. Up till now, one can observe that two nodes are contradicting each other, Node X thinks T_2 is problematic yet Node U claims that T_2 is correct, so one of two nodes must be malicious or wrong. To solve this issue, nodes post T_2 into the main blockchain and a proof of the state after executing T_2. This mechanism if utilized for outsourcing computation to untrusted nodes.

19.3 Networking layer of Bitcoin: Broadcasting

In network layer/peer-to-peer layer of Bitcoin, broadcasting is defined as a measure of delivering blocks to everyone in the network. Broadcasting can help every node in the system update its own blockchain and continue to function without fail. Moreover, it is expected that the network

protocol for a blockchain system is able to minimize the updating latency, maximize throughput, provide privacy and security. The simplest protocol is to let each node connect to a fixed number of other nodes. Next, 5 network protocol are introduced as a response to the above 4 goals for the network protocol of the blockchain system.

19.3.1 Compact Blocks

Instead of sending the entire block to neighbours, the hashes of transactions plus the block header are sent to neighbours, which means that the already received transactions do not need re-downloading. Extreme thin blocks and Graphene take one step further by utilizing compressed data structure for sending the transactions. Velocity, as another approach, sends coded data using coding theory.

19.3.2 Reduce Processing Latency

If the block inter-arrival rate is faster than the processing rate, then it will impose problems. Network latency is equal to the propagation latency + download latency + processing latency. The processing latency is not overcome by compact blocks. One idea to cope with this is sub-blocks, in which miner releases unfinished hashes that has not met the valid block requirement. Sub-block method is a proactive method that expedites the verification of the future reception of blocks. In other words, the inter-arrival time slot/ empty cycle between two arriving blocks is utilized to enhance the efficiency.

19.3.3 Network Topology Improvement

Deploy the trustworthy nodes run by operators and transmission through such nodes is called cut-through routing which does not require any verification. However, this may incur some question since this network topology more or less brings a certain degree of centralization to blockchain system.

19.3.4 Perigee

Perigee is a network protocol based on multi-arm bandit which learns based on the past information and the random exploring method. This method is similar to the ϵ-greedy search method of off-policy in deep reinforcement learning [3]. Two benchmarks are introduced as follows: In

Fig. 19.4 (a), we have a random connection from (0,0) to (1.0,1.0); In Fig 19.5, we have a geometric method which aims to minimize the Euclidean distance between two points. Both methods are not be good enough. The first method is complete random, so it is the worst obviously. The second method minimize the physical distance, however, the processing and propagation delay is not optimized. In Fig. 19.5, perigee is shown to outperform the current existing methods since it jointly consider the propagation delay, processing delay and etc. The details can be found in [2].

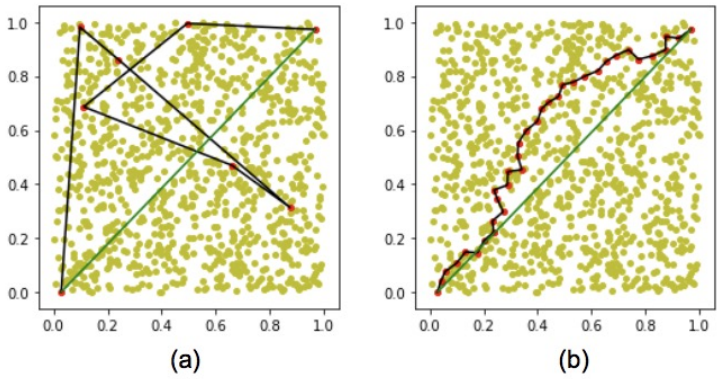


Figure 19.4: Example of 1000 nodes embedded randomly within a unit-square. (a) If nodes are interconnected according to a random topology, the shortest path between two points can be much longer than the Euclidean distance between the points. (b) If nodes are interconnected using a carefully designed topology, significantly better paths, with length close to the Euclidean distance, are possible.

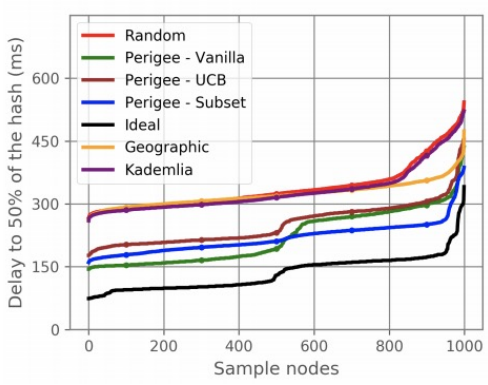


Figure 19.5: Perigee Performance and Comparison with Other Methods

19.3.5 Dandelion

Suppose Node 1 in red is trying to pass its public key into the network of nodes, shown in Fig. 19.6. If Node A, B, C, and D work cooperatively, based on the time stamp and the latency it takes for 4 nodes to receive that public key, they can deduce that the public key is very likely to be initiated from Node 1 in red. This is similar to the utilization of triangulation to locate the origin of the wave front. Hence, the leakage of privacy and the broadcast is more or less insecure. The

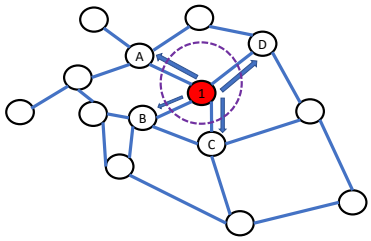


Figure 19.6: Unsafe Protocol

Dandelion protocol [1] improves the privacy and hide the public sender in the network shown in Fig. 19.7. The node 1 in red does not pass the public key in an isotropic way but only to one node at a time, therefore, the public key experiences few steps of random walk and reaches node 2 in red. After that, the public starts propagating in the isotropic way. In such way, node B, C, D cannot deduce the origin of the public key, the best they can refer from the latency and time stamp is to trace back to node 2 in red, which largely improves the privacy of the public key initiator.

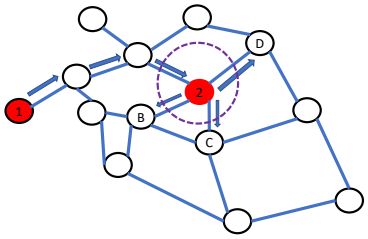


Figure 19.7: Safe Protocol

References

[1] Shaileshh Bojja Venkatakrishnan, Giulia Fanti, and Pramod Viswanath. Dandelion: Redesigning the bitcoin network for anonymity. *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, 1(1):1–34, 2017.

[2] Yifan Mao, Soubhik Deb, Shaileshh Bojja Venkatakrishnan, Sreeram Kannan, and Kannan Srinivasan. Perigee: Efficient peer-to-peer network design for blockchains. In *Proceedings of the 39th Symposium on Principles of Distributed Computing*, pages 428–437, 2020.

[3] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.