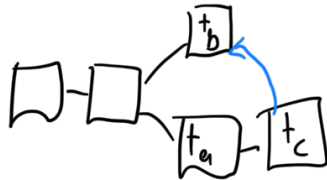


More uncles, bitcoinNG, and sharding

More uncles



Idea: Execute transactions from uncles.

$t_a \leadsto t_b \leadsto t_c$
↑
only execute t_b
if not conflicting

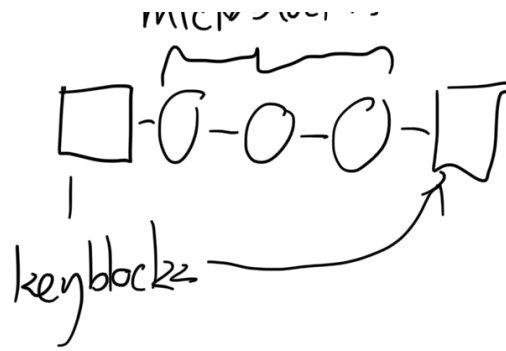
not done in Eth.

Problem

Mostly uncles will include the same transactions

Bitcoin NG

mini-blocks



Key blocks

safety

- No transactions \rightarrow small \rightarrow small network delay δ
- PoW
- seldom \rightarrow difficult PoW \rightarrow small $p \rightarrow$ few forks
- Pk of the new "leader"

Microblocks

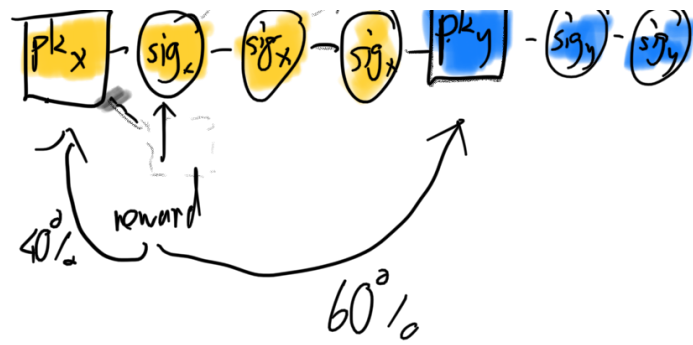
throughput

- No PoW
- Signed by the last "leader"
- Transactions
- frequent
- give reward

safety

Use longest chain rule on key blocks





Possible attacks

1) $\begin{matrix} 80\% & 20\% \\ \curvearrowleft & \curvearrowright \end{matrix}$



2) $\begin{matrix} 20\% & 80\% \\ \curvearrowleft & \curvearrowright \end{matrix}$



Problem: Leader is a target for attack.
e.g. DDOS

Sharding

Shard: Subsystem with fraction of the state.

Prob. limit - linear scaling

10/11/19 (1) (2) (3) (4) (5) (6) (7) (8) (9) (10) (11) (12) (13) (14) (15) (16) (17) (18) (19) (20) (21) (22) (23) (24) (25) (26) (27) (28) (29) (30) (31) (32) (33) (34) (35) (36) (37) (38) (39) (40) (41) (42) (43) (44) (45) (46) (47) (48) (49) (50) (51) (52) (53) (54) (55) (56) (57) (58) (59) (60) (61) (62) (63) (64) (65) (66) (67) (68) (69) (70) (71) (72) (73) (74) (75) (76) (77) (78) (79) (80) (81) (82) (83) (84) (85) (86) (87) (88) (89) (90) (91) (92) (93) (94) (95) (96) (97) (98) (99) (100)

Bank



Accounts: A-M



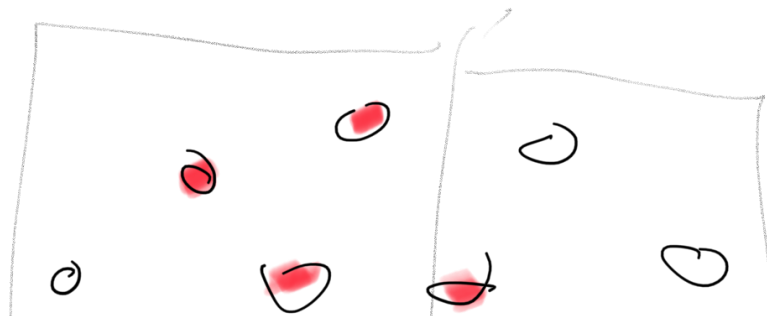
Account: N-Z

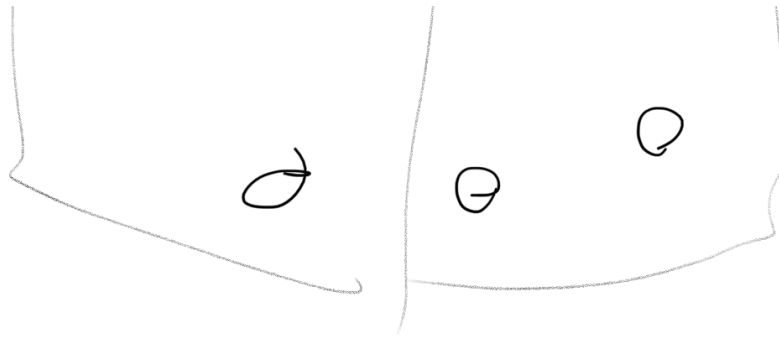
Problems

A: How to divide state?

B: Cross shard transactions?

C: Avoid mining power dilution





A solution:

Hashing ranges

B: Cross shard protocol

