Algorand: Scaling Byzantine Agreements for Cryptocurrencies

Yossi Gilad, Rotem Hemo, Silvio Micali, Georgios Vlachos, Nickolai Zeldovich MIT CSAIL

SOSP’17

Context Introduction

- Cryptographic currencies
- Avoiding centralized authorities
- Trade-off between latency and confidence
- Double spending problem

Nakamoto consensus & Proof of Work

$$\begin{tikzcd}
B_0 \arrow[r] & B_1 \arrow[r] & B_2 \arrow[r] \arrow[dr] & B_3 \arrow[r] & B_4 \\
& & & B_3' \arrow[r] & B_4'
\end{tikzcd}$$

Nakamoto consensus & Proof of Work

$$\begin{tikzcd}
B_0 \arrow[r] & B_1 \arrow[r] & B_2 \arrow[r] \arrow[dr] & B_3 \arrow[r] & B_4 & \\
 & & & B_3' \arrow[r] & B_4' \arrow[r] & B_5'
\end{tikzcd}$$

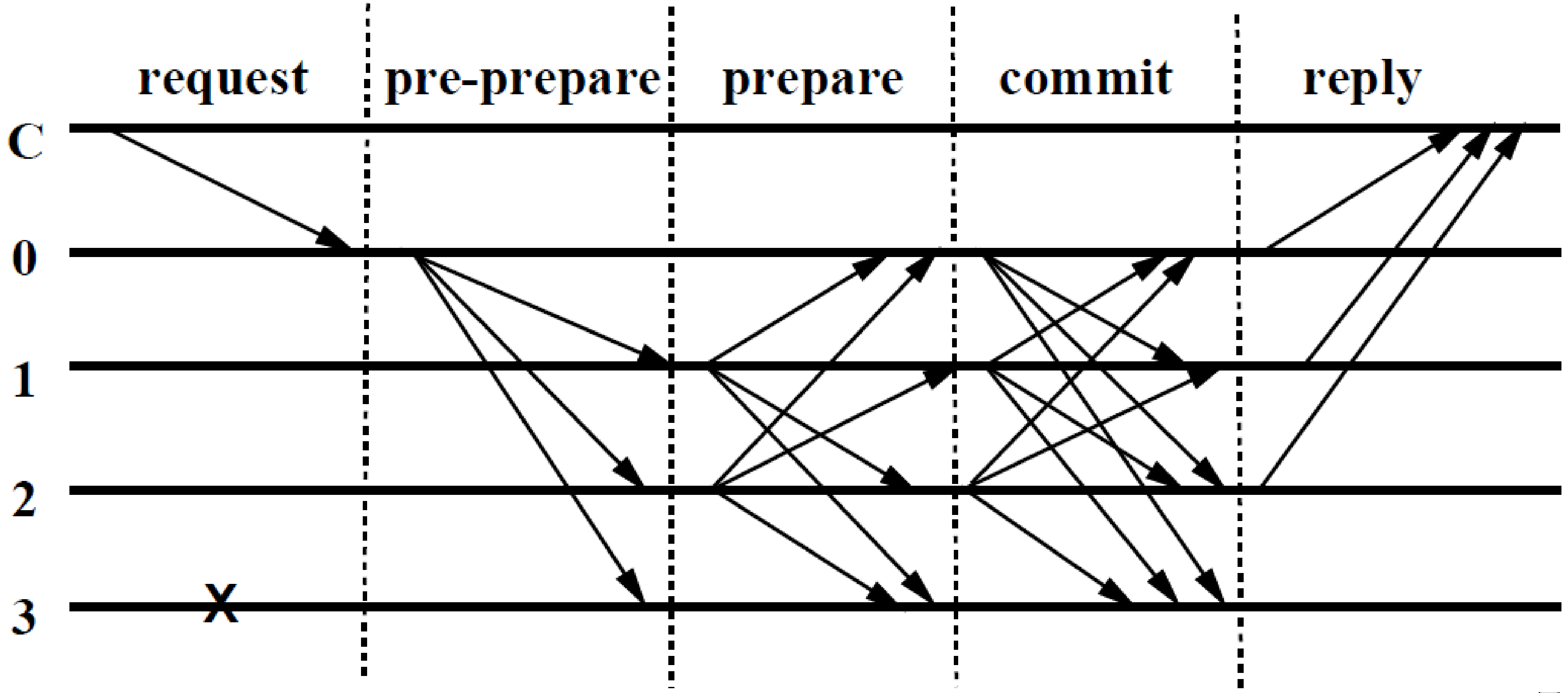
Nakamoto consensus & Proof of Work

- No confident commit
- Possible forks
- Latency problem
- Scalabillity

Byzantine Consensus

- Predefined set of servers
- Denial of service attack
- All to All communication
  - Bad Scalabillity

Example: PBFT



Algorand

- New cryptocurrency
- Confirmation in order of minute
- Scallable (No all to all communication)

Algorand: Network structure

- Dynamic size network
- Scallable (No all to all communication)
- No predefined set of committee

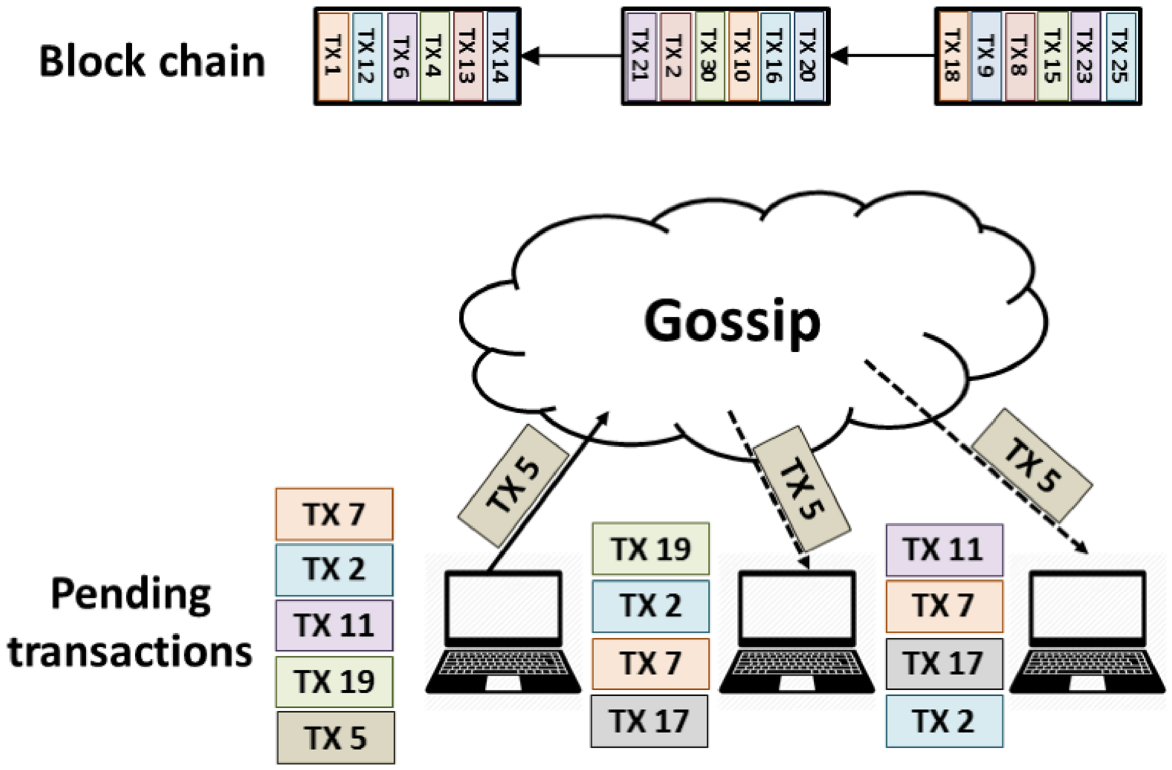
Algorand: BA*

- Proof of Stake
  - Fraction of the money held by honest users is at least a constant greater than 2/3.
- Confirmation in order of minute
- No predefined set of committee

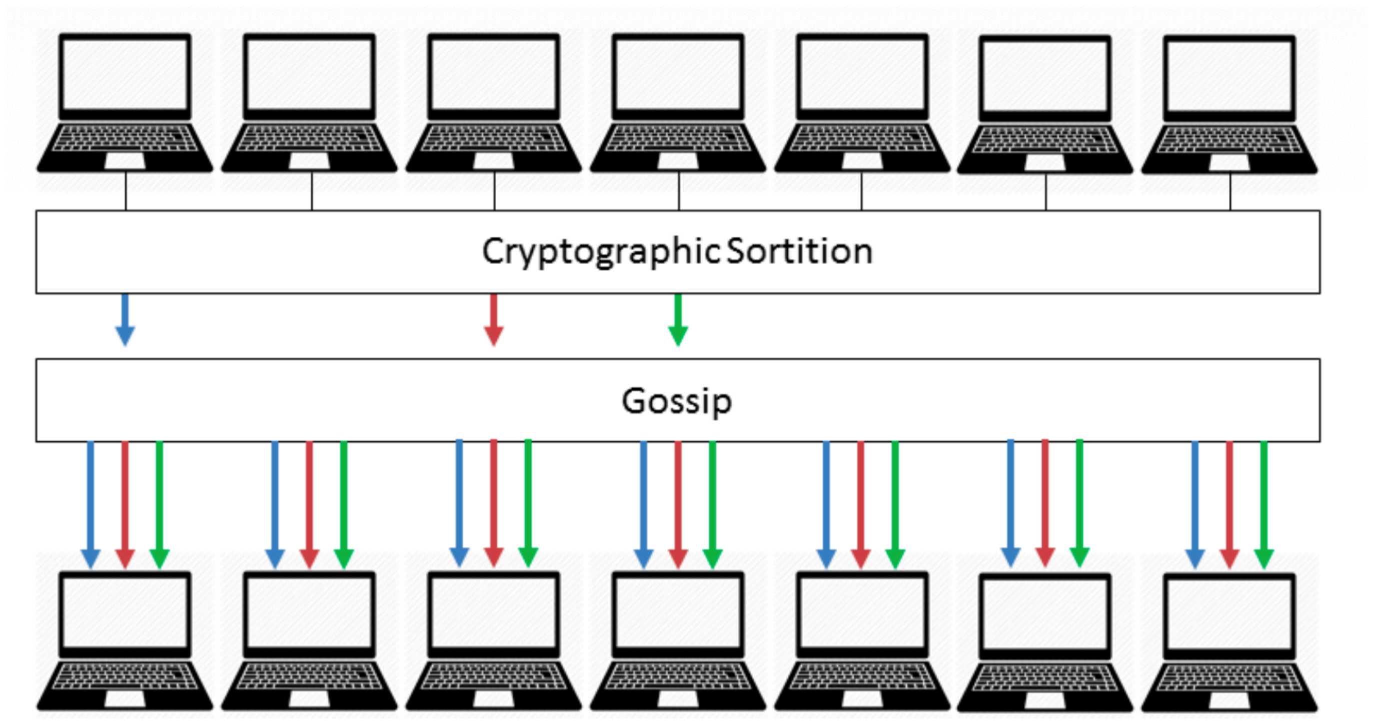
Algorand: Key components

1. Gossip Network
2. Cryptographic sortition (for choosing small committee)
3. BA*

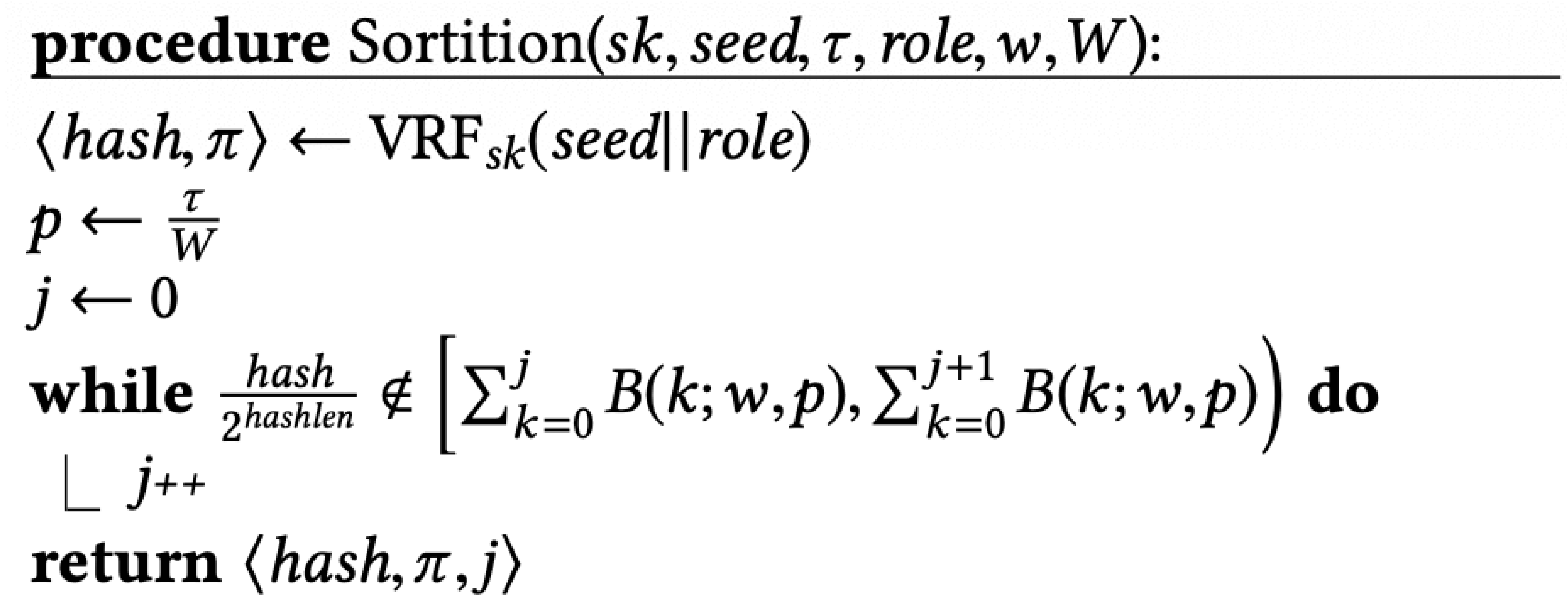
Algorand: Gossip



Algorand: Block Proposal

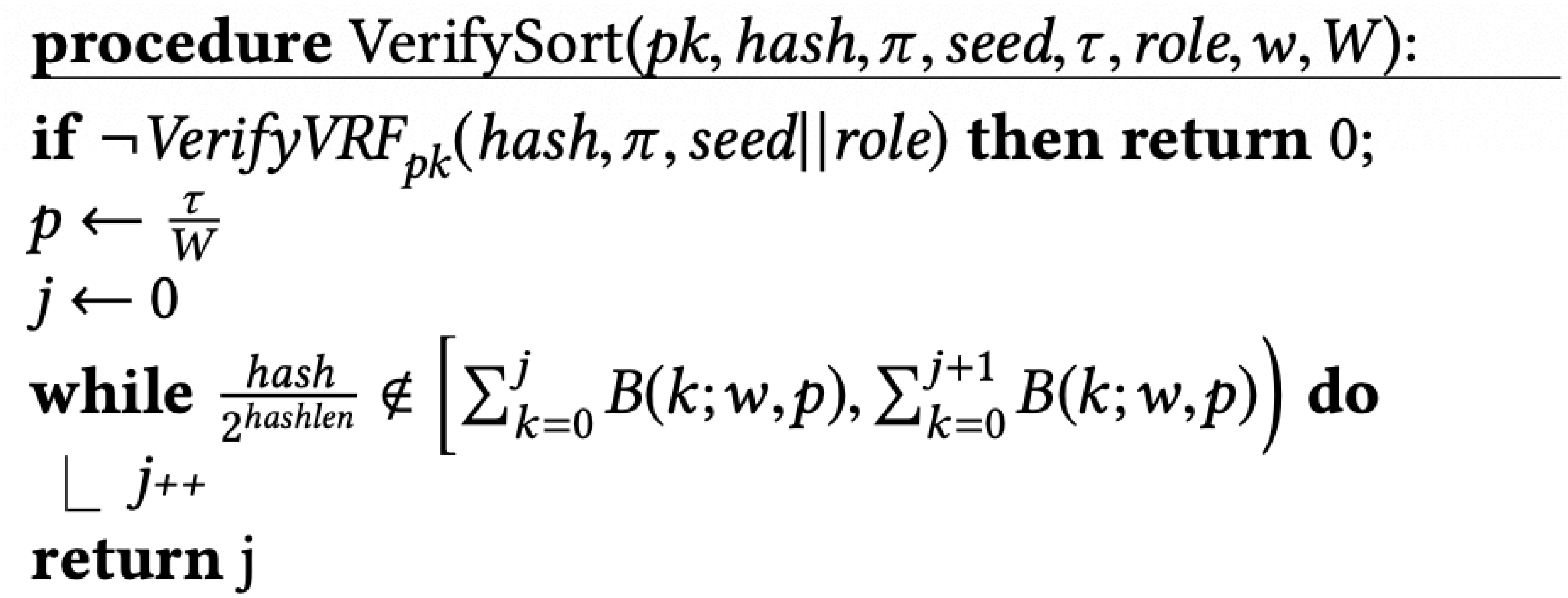


Algorand: Sortition



```
**procedure** Sortition($sk, seed, \tau, role, w, W$):
$\langle hash, \pi \rangle \leftarrow \mathrm{VRF}_{sk}(seed||role)$
$p \leftarrow \frac{\tau}{W}$
$j \leftarrow 0$
**while** $\frac{hash}{2^{hashlen}} \notin \left[\sum_{k=0}^{j} B(k; w, p), \sum_{k=0}^{j+1} B(k; w, p)\right)$ **do**
    $j$++
**return** $\langle hash, \pi, j \rangle$
```

Algorand: Sortition Verification



```
procedure VerifySort($pk, hash, \pi, seed, \tau, role, w, W$):
if $\neg VerifyVRF_{pk}(hash, \pi, seed||role)$ then return 0;
$p \leftarrow \frac{\tau}{W}$
$j \leftarrow 0$
while $\frac{hash}{2^{hashlen}} \notin \left[\sum_{k=0}^{j} B(k; w, p), \sum_{k=0}^{j+1} B(k; w, p)\right)$ do
    $j$++
return j
```

Algorand: BA*

```
procedure $BA\star(ctx, round, block)$:
$hblock \leftarrow \text{Reduction}(ctx, round, H(block))$
$hblock_\star \leftarrow \text{Binary}BA\star(ctx, round, hblock)$
// Check if we reached “final” or “tentative” consensus
$r \leftarrow \text{CountVotes}(ctx, round, \textsc{final}, T_{\textsc{final}}, \tau_{\textsc{final}}, \lambda_{\textsc{step}})$
if $hblock_\star = r$ then
    return $\langle \textsc{final}, \text{BlockOfHash}(hblock_\star) \rangle$
else
    return $\langle \textsc{tentative}, \text{BlockOfHash}(hblock_\star) \rangle$
```

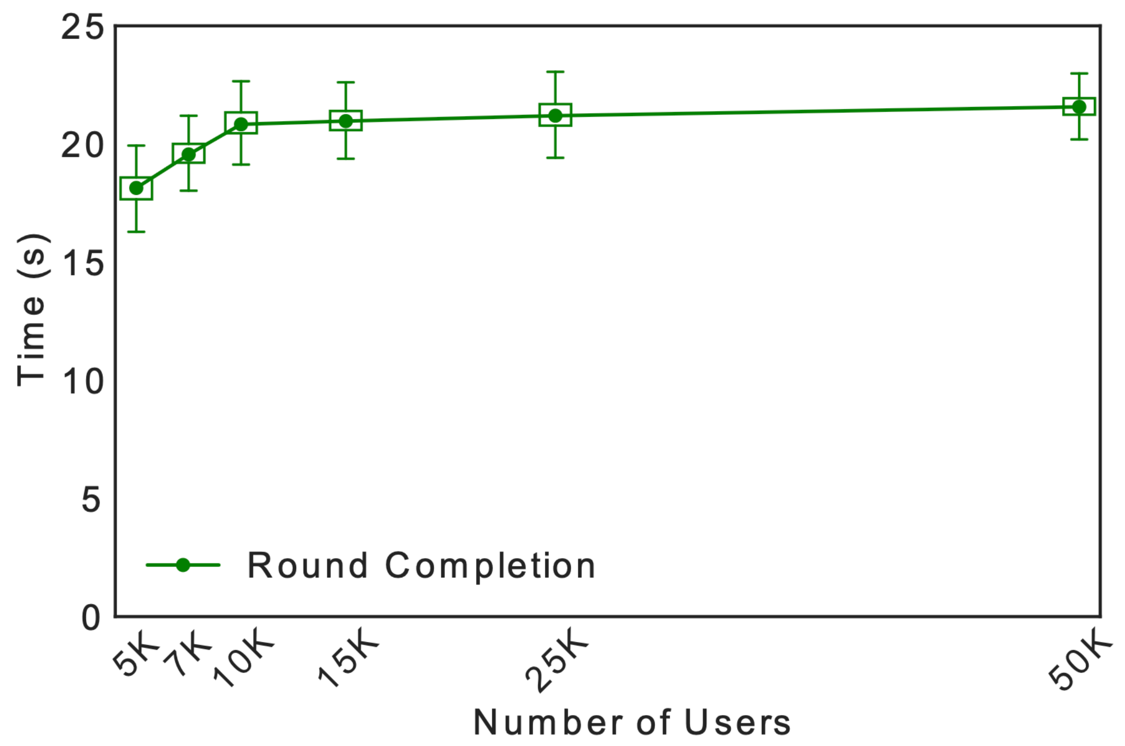


Figure 5: Latency for one round of Algorand, with 5,000 to 50,000 users.

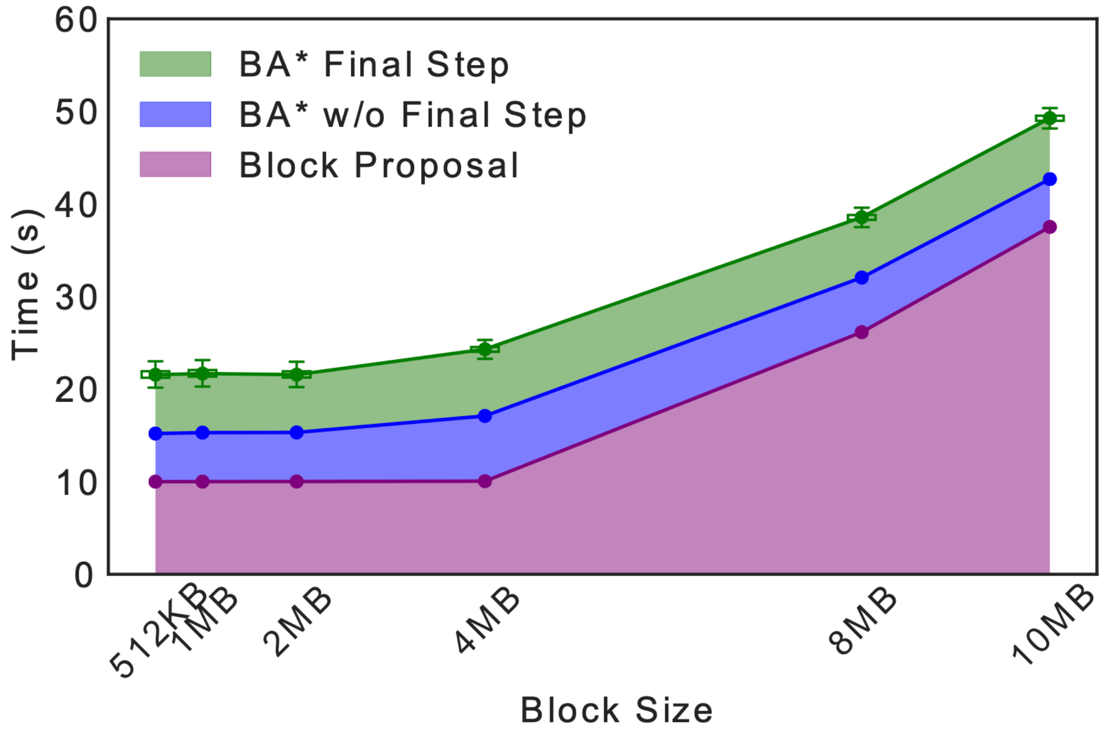
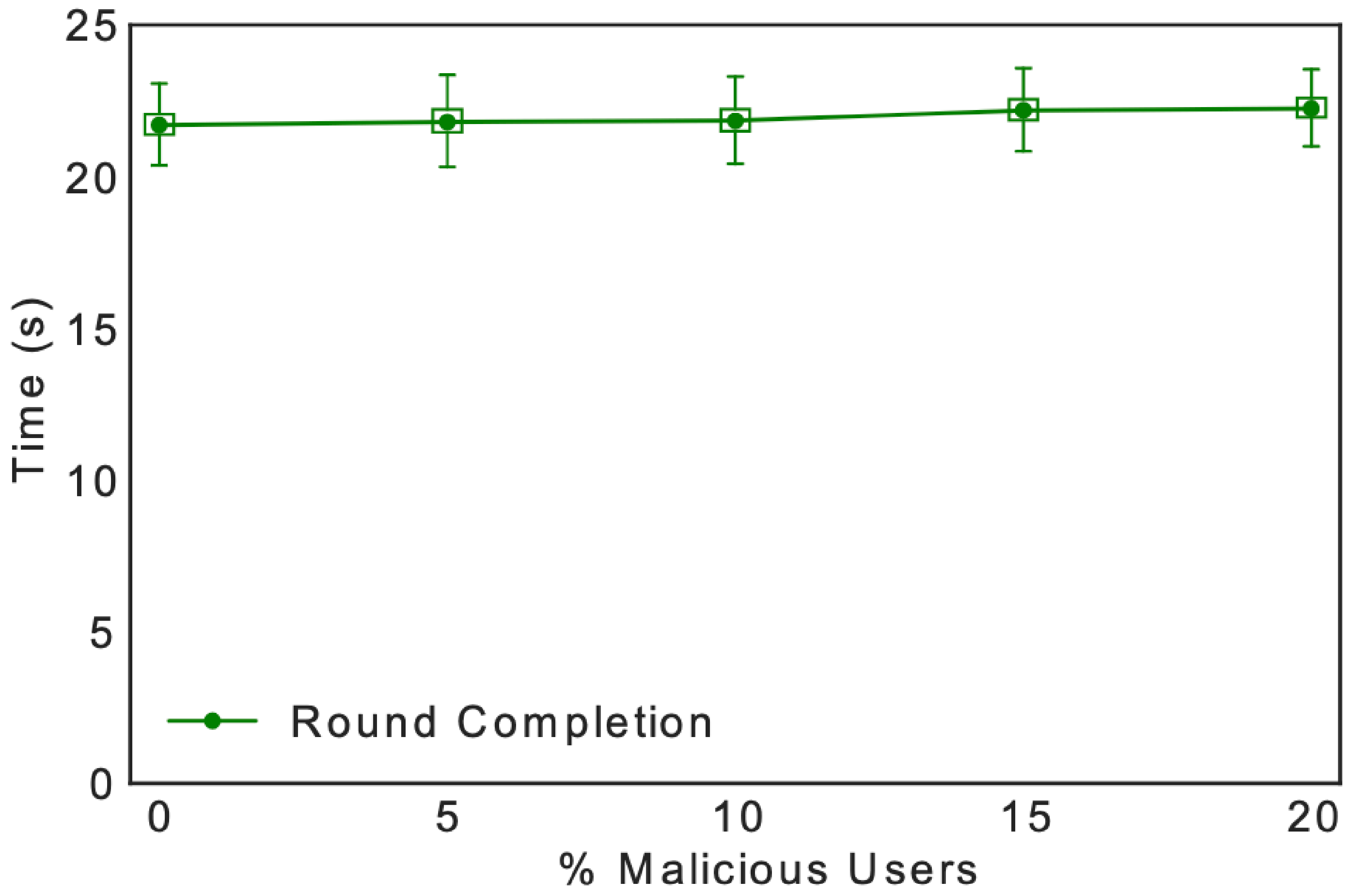


Figure 7: Latency for one round of Algorand as a function of the block size.


Time (s)
25
20
15
10
5
0
0
5
10
15
20
% Malicious Users
Round Completion