Algorand: (Another) Better Bitcoin?

Based on:
Algorand: Scaling Byzantine Agreements for Cryptocurrencies, by Yossi Gilad et. al.

Presented by:
Guozhen Li
ECS 265 Distributed Database Systems, Fall 2018, UC Davis
Nov 27, 2018

What's bad about Bitcoin

- **Wastes** electricity
- **Not really distributed**: computing power, thus decision power, (eventually) controlled by a few (~5) big mining companies
- **Vulnerable**: the big miners are known to the world & they have low profit margins → easy to corrupt
- **Scalability** is questionable
- **Ambiguity**: forks can form
- **Slow**: transaction takes ~1hr to confirm

Algorand vs. Bitcoin

	Bitcoin	Algorand
Who decides what value to agree on	One node that solves a complex puzzle fastest	Majority vote from a randomly selected committee
Main assumption	Majority of computing power is honest	Majority of funds are held by honest users
Computation workload on a node	Heavy: find a needle in a haystack	Light: add, count, compare, sign, verify
True decentralization?	Not really. Faster nodes have more power.	Yes (kinda). Everyone has a chance to vote.

Adding a Block in Algrand (when all goes well)

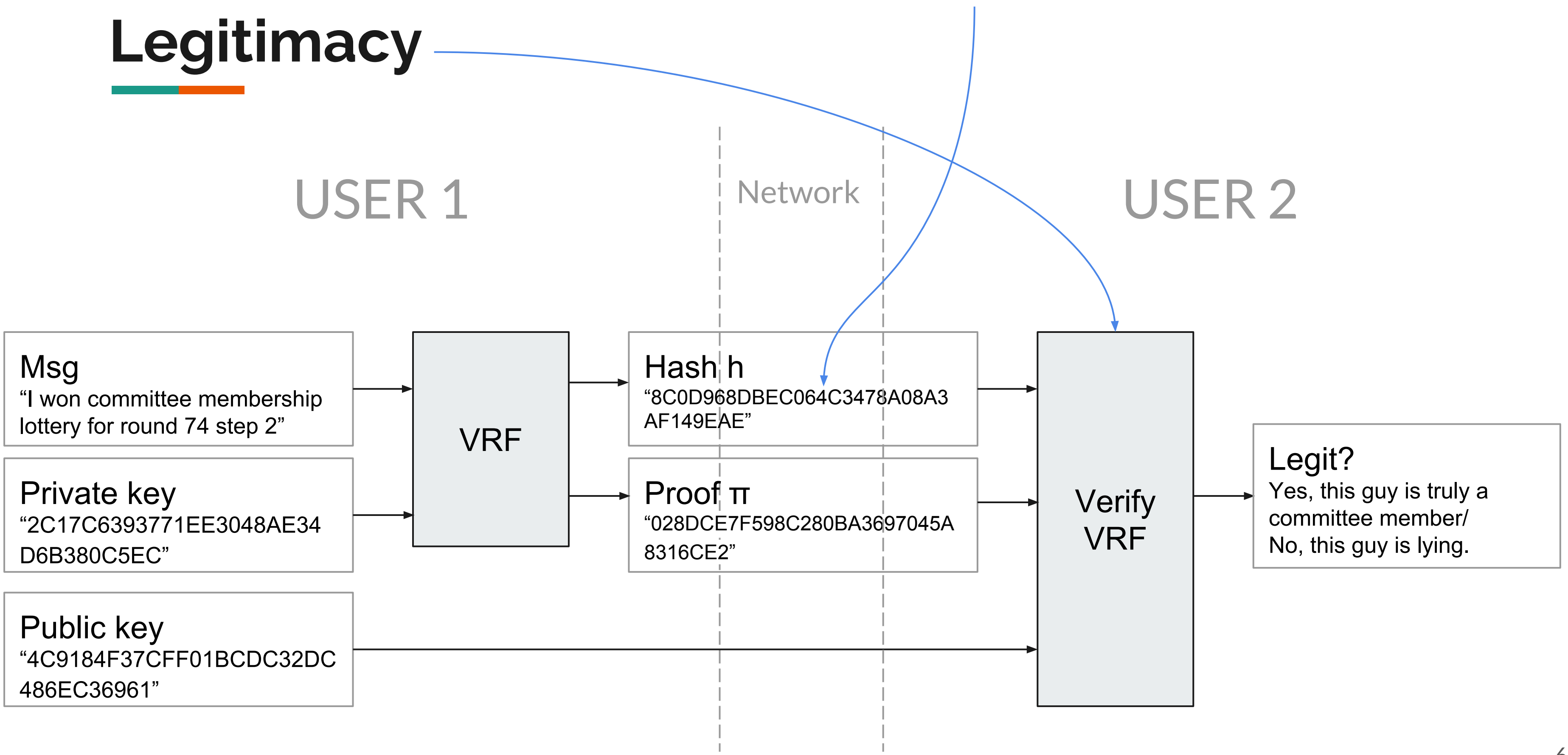
1. A **random group** of users (e.g. 26 users) each proposes a block based on payments it has observed from gossips, then broadcast its proposal to all users via gossiping.
2. A **random committee** (e.g. 1000 users) each collects proposals from **legit proposers**, and broadcast that it votes to the one proposal it heard often enough.
3. A **different random committee** (e.g. 1000 users) each counts **legit votes** from the previous committee. For each of them, if one proposal is found to win majority (e.g. over ⅔ of previous committee) votes, that committee member accepts that proposal, and gossip “I accept block X”.
4. For all users, when they hear enough **legit committee members** say “I accept block X”, they also accepts block X. Thus the network reaches consensus

VRF: The Guarantee for Randomness and Legitimacy

VRF = verifiable random function

- Everyone runs a “lottery” on its own
- The lottery generates a “winning ticket” and a “proof”, if one wins a role (e.g. proposer, committee)
- Everyone signs the winning ticket with its private key, and gossips out the signed winning ticket with the proof
- Everyone can verify everyone else’s “signed winning ticket + proof” pair to determine legitimacy
- Everyone only takes into account votes from verifiable messages

VRF: The Guarantee for Randomness and Legitimacy



Algorand in More Details

(Sections 5-7)

CRYPTOGRAPHIC SORTITION - committee election/lottery

BLOCK PROPOSAL

BA★

Algorand in More Details: BA★

Two phases in BA★:

1. Reduction()
   "Everyone choose one of {proposal#56346, proposal#12059, empty_block} to pass to BinaryBA★()"

2. BinaryBA★()
   "Everyone choose one of {proposal_from_reduction, empty_block} as your final choice"

After these two phases, everyone counts other users' final choices from gossips.
If your *proposal_from_reduction* receives enough votes, you accept it as a *final* block.
If your *proposal_from_reduction* does not receive enough votes, you mark it as a *tentative* block.

Algorand in More Details: BA★::Reduction()

Reduction(ctx,round,**hblock**):

CommitteeVote(ctx, round, **REDUCTION_ONE**, τstep, **hblock**)	I vote for proposal#12059 in poll REDUCTION_ONE for round 74
hblock1←CountVotes(ctx,round,**REDUCTION_ONE**,Tstep,τstep,λblock+λstep)	Which proposal is the most popular in poll REDUCTION_ONE?
empty_hash ← H(Empty(round, H(ctx.last_block)))	Prepare hash of an empty block, just in case things go wrong.
if **hblock1** = TIMEOUT then	If (from what I head) no proposal wins majority votes from committee
CommitteeVote(ctx, round, **REDUCTION_TWO**, τstep, **empty_hash**)	I vote for empty_block in poll REDUCTION_TWO of round 74.
else	If (from what I heard) some proposal wins majority votes
CommitteeVote(ctx, round, **REDUCTION_TWO**, τstep, **hblock1**)	I vote for that proposal in poll REDUCTION_TWO of round 74.
hblock2 ←CountVotes(ctx,round,**REDUCTION_TWO**,Tstep,τstep,λstep)	Which proposal is the most popular in poll REDUCTION_TWO?
if **hblock2** = TIMEOUT then return **empty_hash**;	If no proposal is popular enough, I pass empty_block to my BinaryBA★(
else return **hblock2**;	If some proposal is popular enough, I pass that to my BinaryBA★()

Algorand in More Details: BA★::BinaryBA★()

Keep doing 3 things:

CommitteeVote(ctx, $round$, $step$, τ_{STEP}, r)
$r \leftarrow$ CountVotes($ctx, round, step, T_{\text{STEP}}, \tau_{\text{STEP}}, \lambda_{\text{STEP}}$)
if r = TIMEOUT **then**
  $r \leftarrow block\_hash$
else if $r \neq empty\_hash$ **then**
  for $step < s' \leq step + 3$ **do**
    CommitteeVote(ctx, $round$, s', τ_{STEP}, r)
  if $step$ = 1 **then**
    CommitteeVote(ctx, $round$, FINAL, τ_{FINAL}, r)
  return r
$step$++

CommitteeVote(ctx, $round$, $step$, τ_{STEP}, r)
$r \leftarrow$ CountVotes($ctx, round, step, T_{\text{STEP}}, \tau_{\text{STEP}}, \lambda_{\text{STEP}}$)
if r = TIMEOUT **then**
  $r \leftarrow empty\_hash$
else if $r = empty\_hash$ **then**
  for $step < s' \leq step + 3$ **do**
    CommitteeVote(ctx, $round$, s', τ_{STEP}, r)
  return r
$step$++

CommitteeVote(ctx, $round$, $step$, τ_{STEP}, r)
$r \leftarrow$ CountVotes($ctx, round, step, T_{\text{STEP}}, \tau_{\text{STEP}}, \lambda_{\text{STEP}}$)
if r = TIMEOUT **then**
  if CommonCoin($ctx, round, step, \tau_{\text{STEP}}$) = 0 **then**
    $r \leftarrow block\_hash$
  else
    $r \leftarrow empty\_hash$
$step$++

Gist of Algorand

- Resolve disagreements with many polls
- For each poll, a different random committee show up and "shout out" their choice
- Everyone keeps listening the "shout outs" in the gossips, and decide what to choose in next poll
- VRFs (along with verifier functions) provide:
  - Randomness of whose "shout outs" are counted.
    (If most people are honest, I make good decisions most of the time.)
  - Legitimacy of the messages in gossips.
    (I can verify whether what I hear is truly that person saying a true thing)

Some Critiques of Algorand

- Not tested in any real-world environment
- No source code or binary released to public yet
- No incentives for users to turn on their machines and participate in the consensus protocol
- In its early years, it is easy for an adversary to buy over ⅔ of all funds in the network

References

- Gilad, Yossi, et al. "Algorand: Scaling byzantine agreements for cryptocurrencies." *Proceedings of the 26th Symposium on Operating Systems Principles*. ACM, 2017.
- (Video) “CESC2017 - Silvio Micali - ALGORAND”, uploaded by Blockchain at Berkeley: https://youtu.be/NbnZi9SImYY
- (Video) “What is Algorand?”, uploaded by Jackson Palmer: https://youtu.be/pLCmL7681oU