YOU ONLY SPEAK ONCE: PRIVATE COMPUTING ON PUBLIC BLOCKCHAINS

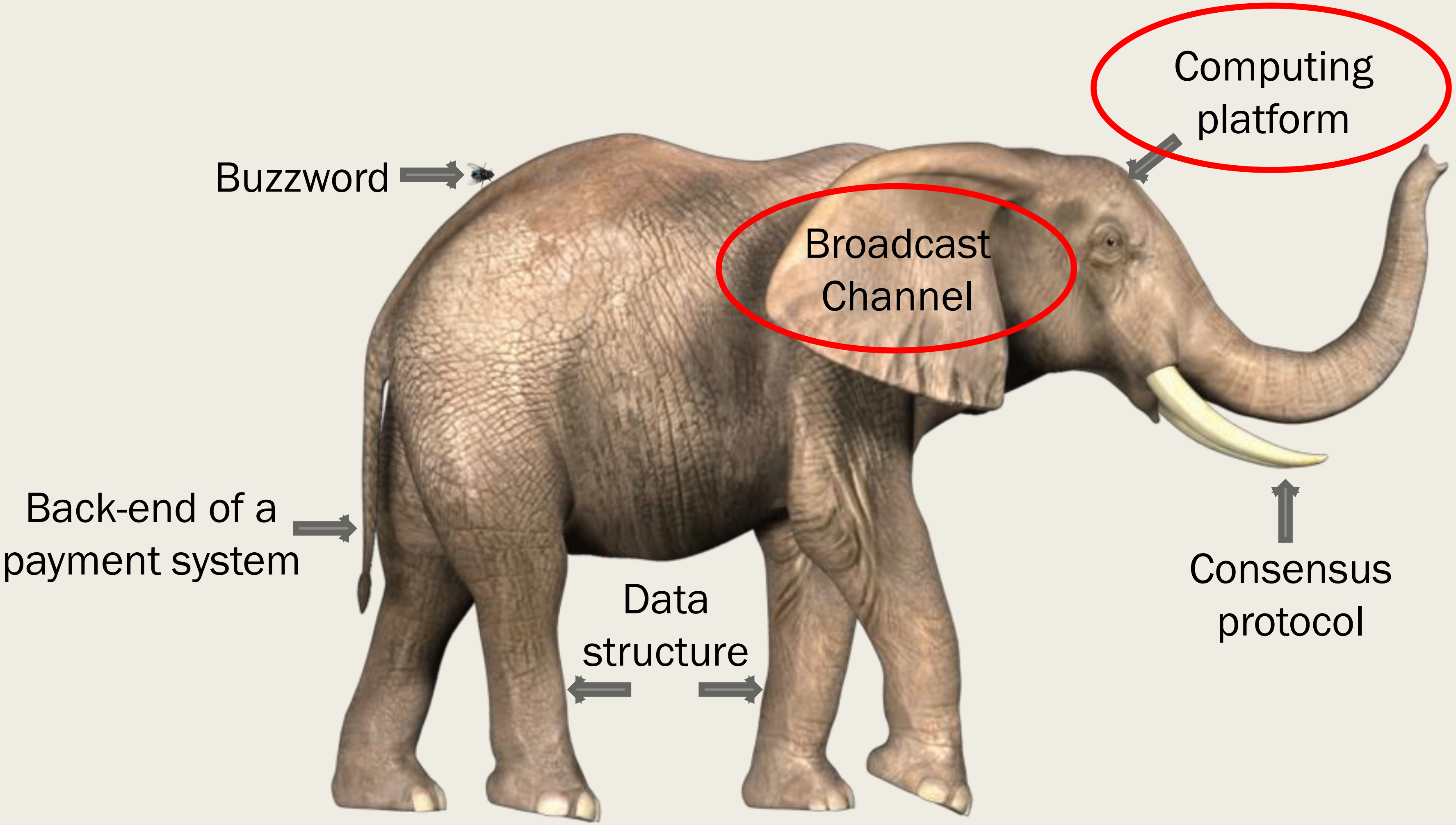
Hugo Krawczyk

Algorand Foundation

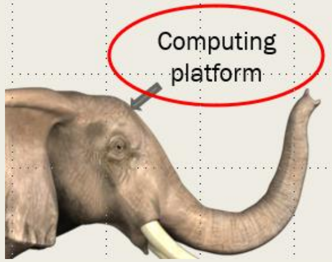
Based on several works and many colleagues
(details at the end)

What is a Blockchain?

- Depending on which part of it you are dealing with



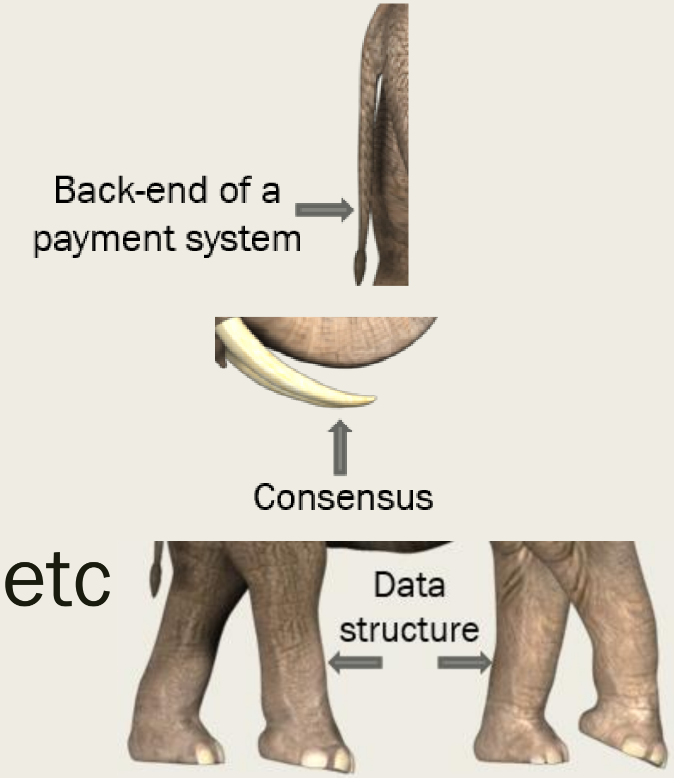
A Public Blockchain



- A distributed network of potentially many nodes
  - *Thousands, maybe even millions*
- Continuously deciding on "things"
  - *These things are called transactions*
  - *Decisions are made by consensus*
  - *Published in blocks, visible and verifiable by all*
- Smart contracts: transaction validity involves running code
  - *Executed publicly, results are agreed by all*

We Will Not Talk About

- Cryptocurrencies
  - *We just assume some way of <u>incentivizing</u> nodes to participate in the system*
- Consensus protocol
  - *We just assume a broadcast channel*
- Implementation issues, data structures, etc
- Details about the chain and the blocks



We Will Not Talk About

- Cryptocurrencies

  ay of incentivizing

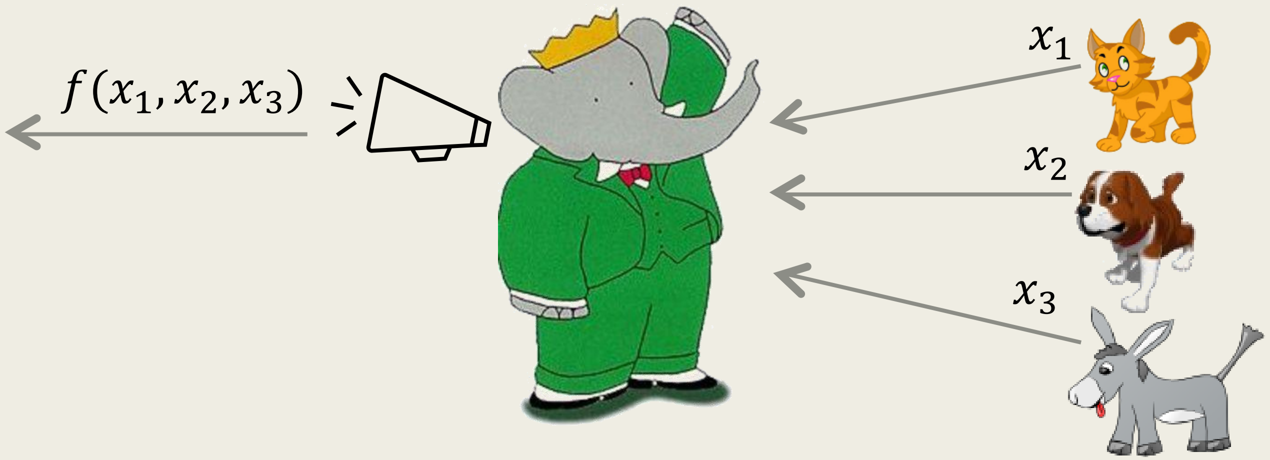
Generality: Blockchain ≈ A large distributed system with a broadcast channel

- mplementation issues, dat

- Details about the chain and the blocks

Public Blockchains as Computing Platforms

- We abstract "Computing platform" as a trusted party



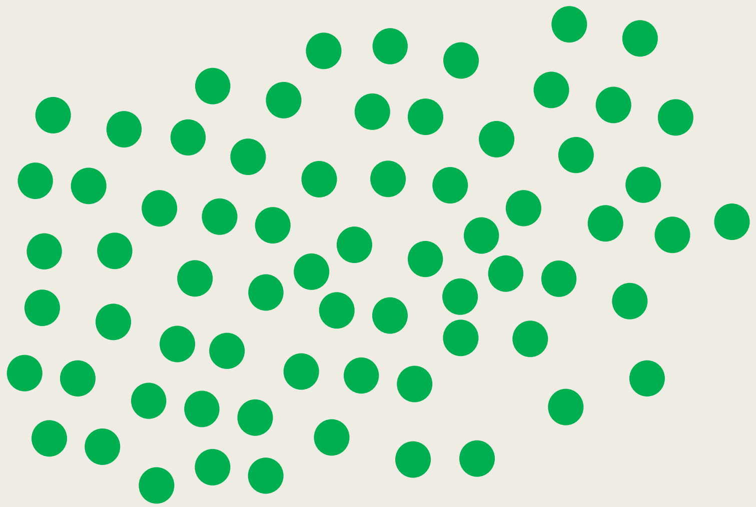
- Can today's public blockchains be trusted parties?
- Not fully…
  - Great for integrity and immutability
  - ***Secrecy is harder***, *this is the focus of our work*

Applications

- Threshold signatures: CA, code signing, notarization
- Key management, secure storage (incl. long-term secrets)
- (Threshold) cryptography as a service: sign, encrypt, O/PRF..
- Threshold FHE (implies threshold obfuscation)
- Secure multiparty computation (MPC)
- Randomness Beacon
- Blockchain checkpoint (and cross chain)
- . . .

Blockchain as Trusted Party

Adversarial Model



- Network of many nodes
- Most of them are honest
  - *Dishonest set can change from one step to the next*
  - *Node can be honest, then become dishonest, later recover and become honest again, etc.*
- "Dishonest" could mean many different things
  - *Fail-stop (e.g., under DoS attack)*
  - *Leaky (follow protocol but attacker know their secrets)*
  - *Malicious (arbitrary behavior)*

Our Goal: Secure Scalable Computation

- Computation should not increase in complexity as more nodes join the system
  - *For us: complexity bottleneck = broadcast bandwidth*
- For scalability: **let a small committee do the work**
  - *E.g., choose a different random committee in every step*
  - *Chosen at random ➔ represents the entire system whp*
  - *In particular, with high probability, the committee has honest majority*

Example: Algorand's "player replaceability"

The Main Technical Challenge

- Some fraction of the nodes can be adversarial
  - *E.g., $f = 25\%$, chosen adaptively by adversary*
- For scalability, communication only by a small committee
  - *Much smaller than an f-fraction of the nodes, e.g., 2%*

→Adversary has enough "budget" to target them all

- **Only if it knows who the committee members are!!**

The Main Technical Challenge

- How can the committee do its job, without revealing to the adversary who they are?
- Even if some committee members are adversarial
- With a *public* broadcast channel as the only means of communication

Introducing: YOSO Protocols

You Only Speak Once

YOSO Model

- Nodes are interchangeable in the eyes of the adversary
  - *Until they send messages*
- A node can monitor communications, do local work
  - *Learns whether it has been chosen for a committee*
  - *But adversary only learns that a node is on the committee when that node sends messages in the protocol*
- To stay anonymous, a node <u>broadcasts just one message</u> as a committee member
  - *After all its work is done, it has no more secrets left, erases state*
  - ***Too late for the attacker to get hold of the node***

YOSO Protocols

- A new formal notion but important examples already exist

- YOSO protocols for Leader Election
  - *Nakamoto consensus (Bitcoin)*
  - *Consensus via cryptographic sortition (Algorand)*

- Committees chosen by a lottery mechanism

Leader election in blockchain as YOSO protocol

- Bitcoin: A “puzzle challenge” announced in each round, the *first to solve* is elected a leader (it chooses next block)
  - *The solution is verifiable by all*
  - *The leader speaks only once: when it announces block (too late for the attacker to corrupt)*

- Algorand: Self selection by sortition
  - *Each party has a pair (sk, pk) for a VRF (Verifiable Random Function)*
  - *A challenge x is broadcast; party with lowest $VRF_{sk}(x)$ is chosen*
  - *VRF result unpredictable but verifiable; leader speaks once*

YOSO Beyond Leader Election?

- Leader-election with Nakamoto/Sortition are examples of ***public roles***
  - *They do not depend on incoming secret communication*
- But we sometimes need also ***secret-state roles***
  - *Ones that depend on* <u>receiving some secrets</u> *over the network before a node can perform its computation*
- Nodes cannot self-select to fill secret-state roles
  - *Roughly: If no one knows that they are selected, then no one can send them the secrets that they need*

YOSO Protocol Specification

- Specified in terms of roles
  - *abstract parties rather than physical ones*
  - *"Player 3 in Round 7", "Share holder 2 of secret 5", …*
- Roles execute actions specified by the protocol
  - *When roles produce output, they erase state and stop*
- Need to decompose protocol into roles that speak only once
  - *Challenging as in most protocols, parties speak multiple times*
  - *Roles replicate themselves for future actions (non-trivial)*

    → Specialized protocols

Role Assignment (to physical machines)

- At execution time, roles are assigned to actual machines
- Assignment done covertly (unpredictably for attacker)

- Typically, assigned machines chosen at random from universe of machines, e.g., blockchain nodes
  - *Assigned machines should learn what role they were assigned (without having to speak themselves)*
  - *No one should learn any other information*

Role/machine secret communication

- How does Machine M1 assigned role R1 communicate with machine M2 assigned role R2?
  - *Think of Role-based encryption (as in identity-based encryption)*
  - *To send m to M2, M1 encrypts m under "R2 key" and broadcasts ciphertext*
  - *The assignment of role R2 to M2 includes the private key needed for R2-decryption*

  (Role-based encryption is a good abstraction, but can use regular public keys too)

YOSO Specification has two components

1. Role assignment protocol (how to assign roles to machines)
2. Role interaction protocol
   - *Specifies roles' actions*
   - *"Role 7 in round 5 reads values broadcast by Role 3 in round 2 and sends their sum to Role 2 in round 8"*

- The two modules may be independent and have multiple independent instantiations

Role Assignment Protocol (assumes PKI)

- Choose a nominating committee
  - *For example, Nakamoto or self-selection as in Algorand (nominators prove they were selected and speak once!)*
- Each N_i in a *nominating committee* $N_1,\ldots,N_n$:
  - *Chooses a party* P_j *from the set of all parties to fulfill Role R*
  - *Chooses a random ephemeral pair* (sk^*, pk^*)
  - *Broadcasts* $(pk^*, Enc_{pk_j}(sk^*))$
- Everyone can communicate with P_j using pk^* w/o knowing who P_j is (pk^* will represent the role R assigned to P_j)
  - *Note: Assumes "anonymous PKE" (ciphertext independent of pk)*

PIR-based Role Assignment

- Above solution can only stand 29% corrupted parties
  - *Assume adversary can controls* f*-fraction of the nodes*
  - *Chosen committee will have* $\approx 2f$*-fraction dishonest nodes*
  - *Needs* $f < 0.29$ *to guarantee honest majority of the chosen committee*
- A better method: Can withstand 49% corruptions
  - *Assignment function computed using YOSO MPC*
  - *Emulates a "Random PIR Selection"*

Example of YOSO Protocol

Proactive Secret Sharing

(basis to threshold cryptography, multi party computation and more)

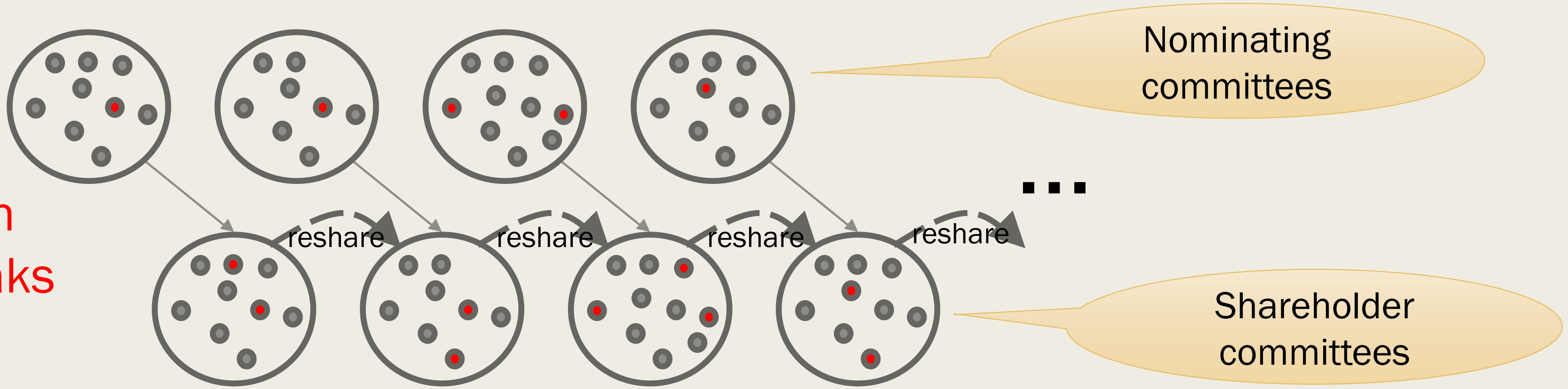
Proactive Secret Sharing

- A secret s is shared among n parties [Shamir79]
  - Every subset of $> {}^{n}/_{2}$ of them can recover s
  - But a subset of $\leq {}^{n}/_{2}$ has no information about s
- Mobile adversary can target many parties over time [OY91]
  - *Eventually it can collect a majority of the parties*
- To Mitigate, refresh shares periodically [CH95, HJKY95,…]
  - *Secret remains hidden if honest majority in each step*
  - *Even if different parties are compromised in different times*

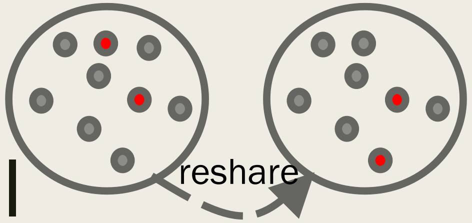
Proactive SS: YOSO Solution Overview

- Secret is shared among a small committee
- Every minute/hour/day, run a re-sharing protocol:
  1. *Nominating committee self-selects, then chooses a fresh random shareholder committee*
  2. *The old shareholder committee reshares the secret to the new one over ephemeral public keys*

While each party speaks only once!



Passing the Secret Between Committees



- We describe a YOSO-style share-refresh protocol
  - *New protocol, using standard techniques*
- Each member of old committee *broadcasts* one message
  - *Fresh shares encrypted under next committee keys*
  - *Including public ZK proofs that re-sharing was done correctly*
    - "The ciphertexts that I sent are consistent with the ciphertexts that I received in the previous step"
    - Broadcast information is linear in the committee-size (independent of the size of the network

Extensions

- Threshold cryptography
  - *Signatures, encryption, PRFs, OPRFs, FHE (→ Obfuscation), …*
- Secure Multi-Party Computation
  - *Information theoretic YOSO MPC (with guaranteed output delivery) and computational role assignment*
  - *Computational YOSO MPC: based on CDN*
  - *More to come*
- Many specific applications

Putting Everything Together

- Theorem: For any function $F(x_1, x_2, \dots)$ and constant $\epsilon > 0$, there is a scalable protocol for securely YOSO-computing F on an N-node network with a broadcast channel
  - *Assuming the adversary controls at most a fraction $\frac{1}{2} - \epsilon$ of the nodes in every time interval*
  - *Each step has communication $\ll N$*

- **In other words: a public blockchain can be a trusted party**

The YOSO Model Beyond Blockchains

- Ephemeral speak-once roles seem a good match for "serverless computing" in the cloud
  - *Can we use YOSO protocols in this context?*
  - *Requires a plausible solution for role assignment*
- Recently Choudhuri et al. described a weaker variant and its use in the context of volunteer-based computation
- Many questions: Models, generalizations, performance optimizations, YOSO designs for specific problems, etc.

THANKS

- *Can a Public Blockchain Keep a Secret?*
  F. Benhamouda, C. Gentry, S. Gorbunov, S. Halevi, H. Krawczyk, C. Lin, T. Rabin, L. Reyzin, TCC 2020, https://ia.cr/2020/464
- *You Only Speak Once: Secure MPC with Stateless Ephemeral Roles*, C. Gentry, S. Halevi, H. Krawczyk, B. Magri, J. B. Nielsen, T. Rabin, S. Yakoubov, Crypto 2021, https://ia.cr/2021/210
- *Random-index PIR with Applications to Large-Scale Secure MPC*, C. Gentry, S. Halevi, B. Magri, J. B. Nielsen, S. Yakoubov, https://ia.cr/2020/1248

Related Work: *Fluid MPC: Secure Multiparty Computation with Dynamic Participants,* Arka Rai Choudhuri and Aarushi Goel and Matthew Green and Abhishek Jain and Gabriel Kaptchuk https://ia.cr/2020/754