Game Theory for Distributed Systems

John P. Conley
Vanderbilt University

and

Geeq.io

Crypto Economics Security Conference

Berkeley

October 2019

Consensus Mechanisms

Consensus mechanisms have the two main jobs:

- Establishing a canonical version of the current state of the data
- Making sure the canonical view is correct

It would be nice if:

- All copies of the database are identical or synchronize quickly
- All copies of the database are available for use
- Altering the data in unauthorized ways is difficult or impossible

CAP Theorem

Unfortunately, the CAP Theorem tells us:

No distributed data store can simultaneously provide more than two out of the following three:

- **Consistency**: Every read receives the most recent write or an error
- **Availability**: Every request receives a (non-error) response – without the guarantee that it contains the most recent write
- **Partition Tolerance**: The system continues to operate despite an arbitrary number of messages being dropped (or delayed) by the network between nodes.

If we could have all three, we would have a canonical view of the state of the database.

FLP Theorem

It would also be desirable if a distributed data system could satisfy:

- **Termination**: Every component will eventually decide on a value.
- **Safety**: Different components will never decide on different values.

Unfortunately, the FLP Theorem (Fischer, Lynch, and Paterson, 1985) tells us that both termination and safety cannot be satisfied in an asynchronous distributed system within a bounded time that is robust to the existence of at least one faulty component.

Byzantine Fault Tolerance

PoW:

- Longest chain rule to get canonicalness.
- Recursive hashing of blocks to make certain rewrites detectable.
- Hashing/nonce search to make rewrites computationally expensive.
- Honesty? 50% BFT.

PoS:

- 2/3 stake weighted voting to get canonicalness.
- Recursive hashing of blocks to make certain rewrites detectable.
- Honesty? 33% BFT.

BFT and Security

Three or four mining pools control a majority of Bitcoin’s hashing power.

It would cost somewhere between $1B and $3B for a bad actor to mount a 51% attack on Bitcoin (and some estimates are lower).

Ethereum and other smaller blockchains would cost much less to attack

PoS is even cheaper to attack and is prone to centralization and control by wealthy agents.

Who would do such a thing?

- USA – Stop tax evaders, money launderers, and criminals.
- China or Russia – Cyber warfare.
- North Korea – Just for fun.
- Canada? - You never know about those guys.

A New Paradigm

Accessibility: There exists a current and correct copy of the ledger upon which non-partitioned users can transact.

It would be great if every node of a distributed data system was current and correct.

This is overkill.

If users can access at least one current and correct version of the data, why do they need others?

A New Paradigm

Provable Honesty: Users with access to a ledger and its supporting transactions can prove whether it is honest or correct in the sense that it has always followed all protocols.

Users must know that the ledger they have access to is correct (no double spending, all signatures valid, etc.).

Any well-designed blockchain that uses a deterministic protocol satisfies this.

Of course, the problem is that separate forks can all be correct but mutually inconsistent.

A New Paradigm

Provable Canonicalness: Users with access to a ledger and its supporting transactions can prove whether it is canonical in the sense that it is authoritative and no other version of the ledger and supporting transactions will ever supersede its authority.

Users must know that the ledger they have access to is canonical and contain finalized transactions

A New Paradigm

PoW ledgers are never provabley canonical

There may be a hidden ledger that is supported by a longer chain, or another fork may come from behind and later become the longest chain.

Users can’t tell anything from the data in any given chain they happen to see.

Longest chain is by definition a measure that depends on external data.

PoS Ledgers have the same problem

If 2/3rds of the nodes are dishonest, they can double sign. That is, they can sign two blocks at each height, hide one until later, and then orphan the first.

Users have no way of knowing that hidden forks don’t exist or won’t be created in future.

A New Paradigm

Proposal:

The design goal of a blockchain protocol should be **Accessibility**, **Provable Honesty**, and **Provable Canonicalness**.

That is, non-partitioned users can transact on a ledger that they know is correct and will never be superseded by another ledger.

Note that this implies that users can also identify and choose not to transact on dishonest or non-canonical ledgers.

This design goal is **not** precluded by the CAP and FLP impossibility theorems.

Honesty and Canonicalness

Key Point: Honesty and canonicalness are logically different concepts.

A ledger supported by the longest chain or endorsed by a qualified majority can be a pack of lies.

An aside:

You may object that if a chain is provabley incorrect, it can't be canonical by definition even if it is the longest chain or has enough votes.

Perhaps, but this does not help users. What if a "canonical" chain contains one trivial error, or nodes have decided to create a fork to roll back "bad transactions"? Do users walk away for their accounts because the chain is provabley incorrect?

Why Byzantine Fault Tolerance?

BFT is a measure of how tolerant a system is to faulty components.

Unfortunately, characterizing robustness in this way tends to make protocol designers think of nodes as parts of a system that either work as expected, or fail, instead of as rational agents with preferences who are capable of doing either depending upon the circumstances.

Honesty is endogenous:

Dishonest ≠ Broken

For the rest of the talk I’m going focus on Honesty/Correctness and leave Accessibility and Canonicalness aside.

Not this again

I can hear you groan: not another economist advocating for game theoretic security guarantees!

Actually, I would argue that game theory does not have a particularly good track record in blockchain.

The design goals of mechanisms that support protocols are simply too weak.

What’s Wrong with Game Theory?

A mechanism is called **incentive-compatible** if it can achieve a desired outcome when each agents acts in their own self-interests. (Some technical details are omitted here.)

For example, honest validation and following protocol rules is a Nash equilibrium in PoS and PoW.

The fatal flaw in this is that there usually **many other equilibria**, most of them bad.

There is no reason to believe that a good equilibrium is more likely than a bad one.

Examples:

- Right side/left side of the road are both Nash equilibria.
- All nodes behaving honestly or all nodes behaving dishonestly are Nash equilibria in PoW and PoS.

What’s Wrong with Game Theory?

Nash is a very weak equilibrium notion. It is only proof against unilateral deviation and depends of the strategy choices of other players.

Dominant Strategy is also very weak. While the best strategy for a single player does not depend on what other players do, coalitions of agents can often do better by collectively changing their strategies (for example, the prisoners’ dilemma).

Are coalitions likely in blockchain?

- Mining pools
- Sybiling
- Collusion among large stakeholders

What does Blockchain need from Game Theory?

A mechanism in which:

- Honesty is the only equilibrium
- The equilibrium is coalition-proof

Coalition-Proof Equilibrium (CPE) extends the idea of Nash equilibrium to coalitional deviations.

A strategy profile is a CPE it no coalition, from single agents, to the grand coalition, have an alternative strategy profile available that leaves all of its members better off. (Again, some technical details are omitted here).

We need a protocol or mechanism that **Uniquely Implements** honesty in CPE

A Unanimity Game

- Agents are offered a chance to play a game in exchange for a one dollar admission fee.
- Each player who pays the fee is sent to a room where a name is written on the wall. Players are asked to write this name on a piece of paper.
- The papers are then gathered and compared. If they all have the same name, then each player is paid two dollars.
- If there is any disagreement about the name, all players get zero (which gives each a net payoff of negative one dollar).

Note that there are many Nash equilibrium including universal truth-telling, a coordinated lie, and discoordination.

This is a feature of most consensus protocols as well.

A Unanimity Game with Auditing

Add the following:

- If all agents write the same name, the named individual gets $1000 (this is like a transaction on a blockchain ledger).
- All agents sign their papers.
- If there is disagreement about the name, then the door to the room is opened, and the name on the wall is read.
- Any player who wrote down the correct name gets $2 plus an equal share of a $1000 bonus.
- Players who wrote down an incorrect name receive nothing.

Equilibrium of a Unanimity Game with Auditing

Truth-telling is the unique coalition-proof equilibrium. (implementation)

Suppose all agents tried to collude and write down one of their own names and then share the $1000 received.

Any single agent who defected and called for an audit would get the $1000 bonus which is **more** than an equal share of the $1000 that the coalition tries to steal.

Knowing that at least one agent will certainly defect, the other agents will abandon the attempt to collude, and so truth-telling is the only equilibrium that remains.

Equilibrium of a Unanimity Game with Auditing

Adapting this basic idea to a blockchain protocol involves a few other things.

- Showing that any equal or unequal division of the any tokens that the grand coalition wants to steal can always be blocked by a sufficient coalition of defecting agents.
- Showing how the provability of honesty allows enforcement of the audits.

These and other details can be found in the following technical paper:

Proof of Honesty:

Coalition-Proof Blockchain Validation without Proof of Work or Stake

https://geeq.io/the-geeq-project-technical-paper/

Conclusion

You can't always get what you want

CAP ⇒ Consistency, Availability, and Partition Resistance are jointly impossible.

FLP ⇒ Termination and Safety are jointly impossible (under reasonable conditions).

BFT ⇒ Certain majorities of non-faulty processes are needed to get consensus.

But if you try sometimes, well you just might find, you get what you need

Accessibility, Provable Honesty and Provable Canonicalness are not impossible.

Implementation of honesty transaction validation and block building by nodes in coalition-proof equilibrium.

Thanks very much!

More details can be found on my webpage:

http://www.jpconley.com

or at the Geeq Project web page:

https://geeq.io/