A Game Theoretic Analysis of Shard-Based Permissionless Blockchains

Hadis Ghafouri

Outlines

- Introduction To Blockchain
- Sharding Protocol
- Rationality
- Game Model
- Nash Equilibrium
- References

Introduction To Blockchain

A blockchain is an append-only, immutable distributed database that records a time-sequenced history of facts called transactions.

Transactions are typically grouped into blocks.

A key aspect of blockchain protocol is the **consensus algorithm** which enables agreement among a network of processors or miners on the state of the blockchain under the assumption that a fraction of them could be malicious or faulty.

Bitcoin's blockchain is **permissionless**, i.e., no trusted infrastructure to establish verifiable identities for processors exists.

The blockchain protocol selects (randomly and in an unbiased fashion) one processor once every 10 minutes on average **(epoch)**, and this selected processor gets the right to commit (or append) a new block onto the blockchain.

Introduction To Blockchain(Cnt'd)

The network (other processors) implicitly **accept** this block by building on top of it in the next epoch or **reject** it by building on top of some other block in the hash-chain.

The Bitcoin protocol uses a Proof-of-Work (PoW) mechanism to select the leader (processor with the right to commit a block) in each epoch in an unbiased fashion.

(PoW) mechanism is a hash puzzle that each processor attempts to solve.

One that succeeds is selected and gets the right to propose the next block.

As PoW involves significant computation, Bitcoin's protocol includes a reward mechanism to incentivize processors to compete (in a fair fashion) and to behave honestly.

One significant shortcoming of Bitcoin's consensus protocol is its **low transaction throughput** and **poor scalability**.

Introduction To Sharding

Sharding proposes to periodically partition the network of processors (in an unbiased fashion) into smaller committees.

each committee processes a disjoint set of transactions (also called a **shard**) in parallel with other committees.

How processors will be incentivized to honestly participate and discharge their committee duties?

As participation in committee tasks (such as transaction validation, signature creation, etc.) impose a cost on processors, it is possible that rational processors may choose not to participate in these tasks (and get away with it as the protocol may still succeed at the end) if their remuneration is not appropriately determined.

if each processor within a committee is equally remunerated, a rational processor may choose to **free-ride**, i.e., get paid without participating in any committee work.

System Model

A network of N processors participating in a public permissionless blockchain.

We assume that all processors are honest, but selfish.

Time is divided into fixed-sized epochs.

The network accepts transactions in blocks, i.e., at the end of each epoch the network accepts and commits a new block of transactions.

Any block B is composed of k disjoint sets of transactions Bi.

Each such disjoint set Bi is referred to as a **shard**.

The number of shards(k) is a variable quantity.

System Model(Cnt’d)

The network determines a binary validation function V.

It takes as an input a transaction (belonging to any shard) and any other data representing the current state of the blockchain.

It outputs whether the input transaction is valid or not.

All processors have access to such a function V.

Sharding is a distributed consensus protocol executed among a set of processors.

It outputs at the end of each epoch a block B containing k disjoint shards Bi.

All honest processors agree on B with a very high probability and all transactions within *B* are valid (i.e., satisfy *V*).

Each committee processes (validates and agrees on) a separate shard(*Bi*).

Sharding Protocol

In each epoch the processors execute the following steps.

1) Committee Formation

First, each processor attempts to generate a publicly verifiable identity by solving some Proof-of-Work (PoW) puzzle.

Each processor uses the solution of a PoW hash puzzle as an identity in that epoch.

Each processor is then assigned to a committee corresponding to its established identity. (say, using the s least significant bits of the identity).

each committee processes a distinct shard based on this s-bit identifier.

Sharding Protocol(Cnt'd)

2) Overlay Setup

Next is the community discovery step where processors discover identities of other processors in their committee by communicating with each other.

The outcome of this step is a fully-connected overlay for each committee in the network.

3) Intra-Committee Consensus

Next, processors run a standard *byzantine agreement protocol* such as *PBFT* within their committees to agree on a set of transactions.

Each committee then sends its consensus set of transactions Bi (or shard) to a final committee for inclusion in the new block B at the end of the current epoch.

In order to be considered by the final committee, each shard Bi needs to be signed by a simple majority, i.e., by at least $\frac{c}{2}+1$ processors for a committee of size c.

Sharding Protocol(Cnt'd)

4) Final Consensus

A final committee then takes the consensus shards (*Bi*) from the previous step and merges these to create a final block B, creates a cryptographic digest or hash of B and broadcasts it to the rest of the network.

5) Randomness Generation for Next Epoch

In the final step of the protocol, the final committee generates a set of random strings and broadcasts it to the network.

These random strings are used by the processors in the identity creation and committee formation tasks of the next epoch.

Processor Costs

We now characterize the costs (including, computation and communication costs) borne by the processors in each epoch due to their participation in the sharding protocol.

The protocol steps in each epoch, as outlined in the previous section, can be grouped into two phases:

1. **Organization phase** (execute steps 1 and 2).
2. **Committee participation phase** (execute steps 3, 4 and 5).

During the organization phase:

- Processors create identities using PoW puzzles.
- Form committees.
- Identify other processors in their committee.

Processor Costs(Cnt'd)

In the committee participation phase:

- Processors validate their respective shards.
- Arrive at an agreement with other committee members.

important points

- the organization phase facilitates the committee participation phase, and is mandatory, i.e., if a processor does not have an identity and gets assigned to a committee, it cannot participate in committee-related tasks.
- the committee participation phase is not mandatory for processors i.e., a processor could choose to create a verifiable identity and be assigned to a committee, but may choose not to participate in tasks such as shard validation and intra-committee consensus.
- we assume that if less than τ processors within a committee of size c do not participate in the committee participation phase, the entire protocol for that epoch fails (no new block is proposed in that epoch).

Processor Costs(Cnt'd)

We characterize the total cost for a processor to participate in an epoch of the sharding protocol based on the cost for executing the above two phases.

Organization phase:

We assume that a processor bears a cost c^m.*(mandatory cost)*

- It is a fixed cost.
- It is independent of the number of transactions processed by the processor.
- Can be approximated using the current difficulty of the PoW puzzle and the average computational power of all the processors.

Committee participation phase:

We assume that a processor bears an *optional cost* c^0 depending on whether the processor fully participates in it or not.

c^0 has two components:

1. a fixed component.
2. a transaction dependent component.

Processor Costs(Cnt'd)

We represent all these per processor fixed costs during the committee participation phase as c^f.

All processors are expected to perform verifying the validity of all transactions (they have received) within their respective shards by using the validation function V.

We represent the cost to validate each transaction using V by c^v.

Total optional cost c_i^0 for a processor p_i:

- $\mathrm{c}_\mathrm{i}^0 = \mathrm{c}^\mathrm{f} + \left|\mathrm{x}_\mathrm{i}^\mathrm{j}\right| \mathrm{c}^\mathrm{v}$

x_i^j is the vector of transactions received and validated by processor p_i.

Average per-processor cost for participation in each epoch : $\mathrm{c}_\mathrm{i}^\mathrm{t} = \mathrm{c}^\mathrm{m} + \mathrm{c}^\mathrm{f} + \left|\mathrm{x}_\mathrm{i}^\mathrm{j}\right| \mathrm{c}^\mathrm{v}$.

Rationality

We assume that processors are honest but selfish.

All processors receive some rewards if the protocol execution in an epoch is successful (block rewards, transaction fees, etc.)

The total benefit or payoff received by processors in each epoch:

- difference between the obtained reward and the spent costs in that epoch.

A selfish (or rational) processor will always choose a protocol participation strategy that improves its benefit or payoff.

If a processor does not execute the organization phase, it does not get any reward as it is not a part of any committee.

A rational processor's strategy could be to execute the organization phase but refrain from the committee participation phase.

The goal of each processor is to maximize its individual payoff(received at the end of each epoch).

We assume that processors do not coordinate in order to jointly maximize their combined utility.

Shard-Based Blockchain Game

A non-cooperative N-Player game model that we refer to as the shard-based blockchain game **G**.

Upon starting an epoch t processors must decide whether to:

- Collaborate with each other.
- Verify transactions.
- Take part in the community participation phase.

We investigate whether block generation can emerge in such a non-cooperative system.

We would like to show that with a uniform distribution of rewards in these protocols, the interactions between processors fall in a category of games, where there exists a **social dilemma** of all-defection behavior.

Game Model

We model shard-based blockchain game G as **a static game** because all processors must choose their strategy **simultaneously**, after they have join their shard.

Processors must decide upon:

- Joining the shards.
- Cooperate and contribute to optional costs or not.

The game G is defined as a triplet ***(P ,S, U).***

P is the set of players, **S** is the set of strategies and **U** is the set of payoff values.

the benefits of successfully adding a block is shared among all processors.

Game Model(Cnt’d)

Players (P)

The set of players **P =** $\{\mathbf{Pi}\}_{(\mathbf{i=1})}^{\mathbf{N}}$ corresponds to the set of processors who have already joined shards in a given epoch time t.

All N processors must have already performed PoW and paid the mandatory costs c^m.

Number of shards in our system model is k.

Each shard has n = N/k committee members.

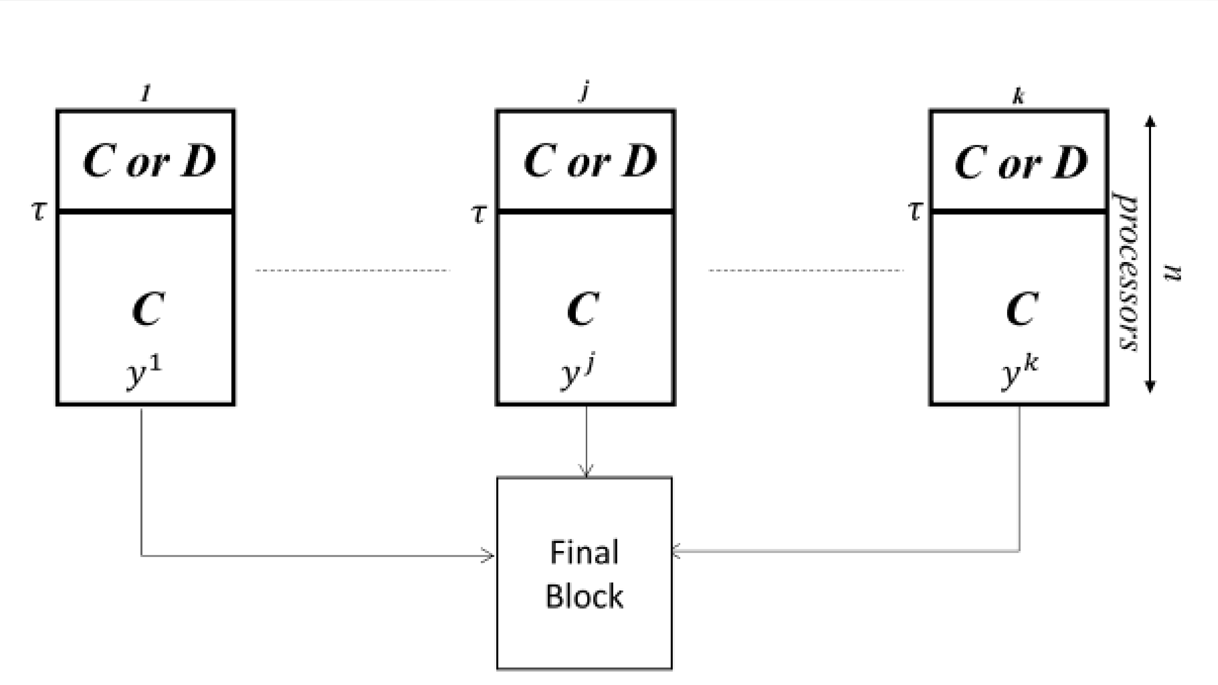
Each processor Pi in shard j receives the vector x_i^j of transactions to verify and participate in the consensus algorithm.

Game Model(Cnt'd)

In each shard at least τ processors among n processors must be cooperative to perform consensus algorithm.

Each Shard j submits the final y^j vector of transactions to make the final block.

y^j represents the result of the consensus algorithm including the list of transactions that would be added to the blockchain by shard j.



Game Model(Cnt'd)

Strategy (S)

Each processor Pi can choose between two moves Si:

- Cooperate C
- Defect D

Set of strategies in this game is S = {C,D}.

The strategy of processor Pi determines whether Pi participates in all optional tasks or not.

If processor Pi plays C:

- it will accept and verify all received transactions.
- It cooperates in all consensus algorithms and incurs cost c^0 for its participation.

If processor Pi plays D :

- it will refuse all transaction verifications.
- It will do nothing during the community participation phase.

Game Model(Cnt'd)

Payoff (U)

After executing the protocol and inserting a new block to the hash-chain at the end of each epoch we assume that the network of participating processors receive two types of rewards.

- Fixed reward for adding a new block, called the block reward(BR).
- Variable reward is the sum of transaction fees of all transactions within the accepted block.

We assume that each transaction includes an average fee r.

The reward or benefit that a given shard j can receive from transaction fees is $r|y^j|$.
Total transaction fee reward due to the appended block in each epoch can be estimated as $TF = r\Sigma_{j=1}^{k}|y^j|$.

Game Model(Cnt'd)

Total cost of cooperation for processor Pi is equal to $\mathbf{c_i^t = c^m + C_i^0 = c^m + c^f + \left|x_i^j\right| \mathit{c}^v}$.

All processors should pay the mandatory costs, i.e., c^m in order to be in a committee and finally receive the reward.

But they can avoid paying optional cost c^o.

we can divide processors into two groups based on whether they contribute to optional tasks:

- Cooperative processors.
- Defective processors.

If one or more shards fail to provide a y^j in an epoch we assume that the network cannot compute and append a new block in that epoch.

Game Model(Cnt'd)

We assume that all processors receive an equal share of profits.

The reward share for each processor is:

$$\frac{BR + r\sum_{i=1}^{k}|y^j|}{N}$$

- If we assume that a processor Pi was cooperative, the payoff of processor Pi in shard j as:

$$u_i^j(C) = b_i - c_i^t = \frac{BR + r\sum_{j=1}^{k}|y^j|}{N} - (c^m + c^f + |x_i^j|c^v)$$

- If Pi is defective:

$$u_i^j(D) = \frac{BR + r\sum_{j=1}^{k}|y^j|}{N} - c^m$$

Nash equilibrium

In a Nash equilibrium strategy profile, none of the players can unilaterally change its strategy to increase its utility.

Any finite game has at least one Nash equilibrium strategy profile.

G is a public good game(PGG).

The system fails to make any new block and remain in the same state if all processors defect initially.

Let us consider the strategy profile where all processors defect and do not pay optional cost c^o after joining to the shards.

We call this strategy profile All -D.

The payoff of each processor i would be then ui $= -c^m$.

In this case none of the processors can unilaterally change its strategy to increase its payoff.

Nash equilibrium(Cnt’d)

The only cooperative processor cannot obtain any reward without the contribution of at least τ-1 other processors in its shard.

new payoff of each processor who deviates: $-c^m - c^f - |x_i^j| c^v$

Which is smaller than $-\mathbf{c}^{\mathbf{m}}$.

Hence, All -D is a Nash equilibrium profile in this game and G is a PGG.

Nash equilibrium(Cnt'd)

In each epoch of this game G with N processors, if rewards are equally shared among all processors, we cannot establish All-Cooperation strategy profile as a Nash equilibrium.

All N processors have already cooperated in transaction verifications (i.e., $All - C$ strategy profile) and paid the optional cost c^o.

If a given processor deviates from the cooperation and plays defection unilaterally:

- Its payoff would be equal to:

$$u_i^j(D) = \frac{BR + r\sum_{j=1}^{k}|y^j|}{N} - c^m$$

- Which is always greater than cooperative payoffs:

$$u_i^j(C) = b_i - c_i^t = \frac{BR + r\sum_{j=1}^{k}|y^j|}{N} - (c^m + c^f + |x_i^j|c^v)$$

Conclusion

Analyze a game theory model of Shard-base permissionless blockchain protocol.

the Nash equilibrium of the game.

If rewards are uniformly distributed among processors, a cooperative equilibrium cannot be enforced in shard-based public permissionless blockchains.

Hence, we can use a new reward sharing approach, which promotes cooperation among processors by providing appropriate incentives.

References

- https://ieeexplore.ieee.org/abstract/document/8558531/