SCALING BLOCKCHAIN CRYPTOCURRENCIES WITH LEVELED ZONES

{ BORYS GURTOVYI AND EDUARD SANOU }
UNIVERSITY OF SOUTHERN CALIFORNIA

INTRODUCTION

Current decentralized cryptocurrencies based on **blockchains suffer from a scalability problem** that makes them unfit to replace centralized payment systems such as VISA and PayPal.

We propose an idea to achieve a higher scalability than current cryptocurrencies by creating a **hierarchy of transaction zones at different levels**, in which the transactions have different properties depending on the smallest level shared zone.
Each level would have a fork of a common blockchain that would be merged with sibling zones (those under the same parent) periodically, with the periodicity being different at each level. The periodicity would be lower on higher level zones.

Transactions within the lowest level zones would be applied and verified immediately whereas those traversing zones would only be applied once the forks of the corresponding zones are merged.
We suggest using 4 levels: global (0), country (-1), state (-2), city (-3).

This design would allow the transaction throughput within a single city to achieve a similar throughput as current blockchains do globally.
In other words, we are **optimizing transaction delays by distance**, which is a common expectation of fiat transactions, while allowing the global transaction throughput to increase significantly.

SCALING WITH LEVELED ZONES

- The global blockchain is started from the genesis block and is **periodically forked according to the next hierarchy**: The global blockchain can only be forked to a country level, so every country will have its own fork. The country level will be forked as well to state/region levels. So every state will have a fork from it's country chain. Finally, the state will be forked to city levels, so every city will have a fork from it's state chain.
- A user assigns a wallet public key to a particular city[a]. This information is stored in the global blockchain. Users don't need to prove their location to create a wallet - it influences only the fork that will validate their transactions. If a user commutes between zones, they can have an account for each zone and split their money among the accounts.
- A wallet can claim to be in a new city any time; this information will propagate to the global blockchain in the same way as transactions.

All transaction in the city will be validated almost immediately (10 seconds) because one particular city level fork is processing only that city's transactions. The majority of fiat transactions happen within one locality - so that is the main goal. In this way we don't have any throughputs problems - this is actually a sharding with a smart choice of zones.

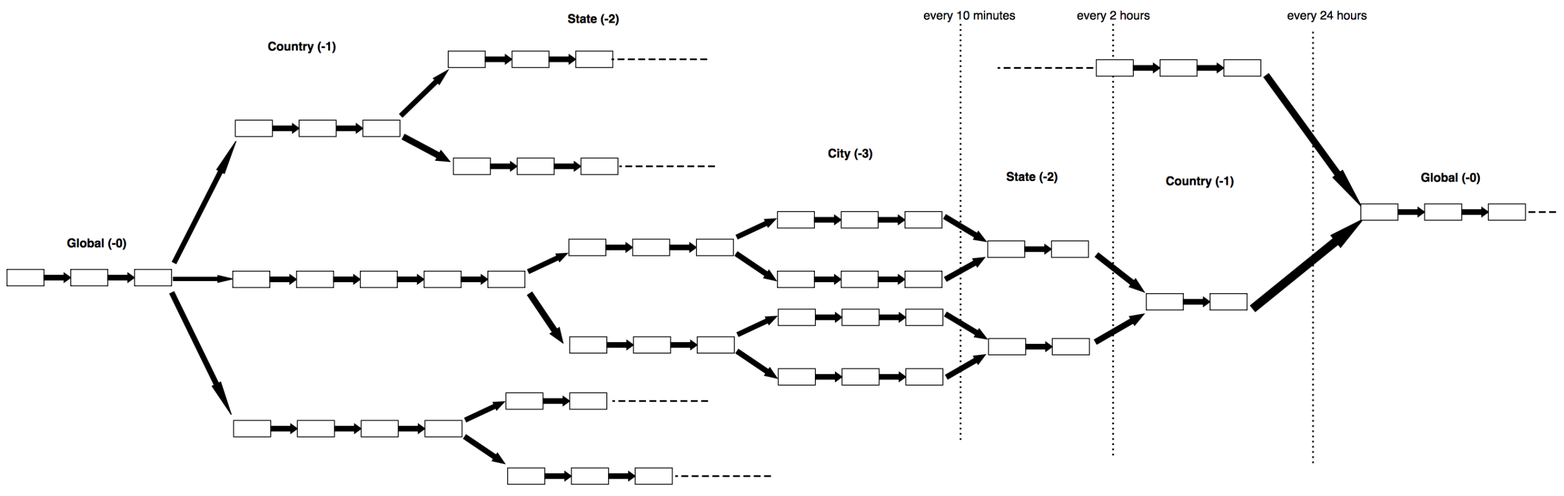
- When user 1 (from city A) sends a cross-city transaction to user 2 (from city B) this transaction is being added to the city A fork as a special "hold" transaction. Basically, user 1 cannot double-spend this money, however user 2 will not verify the transaction until the proper merge happens.
- **Every 10 minutes all cities are being merged within a state**. Because all transactions are verified (at this moment we assume that we can trust other forks were not hijacked) and there is no incoming cross-city transactions (all transactions are either local or "held" output transactions), the merge can be done without any conflicts. Immediately after that a new fork is created for every city and the process starts again.
- **Every 2 hours all state/region forks are synced up** and merged to the appropriate country forks.
- Finally, **every 24 hours all country forks are being merged to the global blockchain**.

Name	Level	periodicity
City	(-3)	10 seconds
State	(-2)	10 minutes
Country	(-1)	2 hours
Global	(0)	24 hours

All exact merging times are scheduled and unchanged.
Each zone only accepts transactions that originate from the same zone.

- In-zone transactions are verified during the merge of the lower level zones (or as they are collected into blocks in the city (-3) level).
- Out-zone transactions are verified when the lowest common upper level zone merges the zones it entails.

Transactions are verified when the forks they belong to are merged into higher zones (or as they are collected into blocks in the city (-3) level).



[a]The locality is called "city" in our system if its population exceeds 1 million people. Other smaller localities will be combined into one unit.

BACKGROUND

All existing blockchain protocols have a serious limitation: scalability. It is impossible to scale popular consensus systems such as Bitcoin because every single node on the network processes every transaction and maintains a copy of the entire state. The benefits of decentralization imply that every node is independent and has equal processing requirements.

With regular database system - the solution would be very easy. We could just add more servers - the number of servers is in direct ratio with the how system can scale. **Adding more computing power may also help but in the decentralized world this means increasing the power of every node.** We cannot force every participant to do so.

In fact, the blockchain actually gets weaker as more nodes are added to its network because of the inter-node latency that logarithmically increases with every additional node. Moreover, requirements for the node increase, and at some point risk of centralization appears (only some part of nodes are able to process the transaction). We end up with the trade off between throughput and decentralization, leading to low throughput in cryptocurrencies.

System	tx/sec	System	tx/sec
Bitcoin	3-4	Paypal	193
Ethereum	20	Visa	2,000
Bitcoin Cash	60		

However this is not enough to meet the needs of fiat currency. Nowadays, Paypal processes 193 transactions per second and Visa 2000 (with a capability of 24,000). **The majority of all transactions are local transaction** (people do fiat currency transactions locally more often than to outside places). This fact pushed us to the idea of scaling blockchain by leveled zones.

CHALLENGES

- **The merging process requires more computational work and bandwidth on higher levels**. Not all nodes may be capable enough to verify merges at every level, Flexibility in the amount of contribution by node may be required.
- **When a node performs a merge, it needs to verify the forks of the sibling zones for correctness**. This could require a lot of computation resources. A possible optimization would be that each merge outputs a compressed form of the result that makes verification at the next level faster.
- **At the lowest level, each zone would have a small number of nodes**. This opens the possibility of an attacker disrupting the zone by registering nodes in it with high validation weight (either computational power or stake). Such disruption could be a denial of service or a denial of selected transactions.
  - A way to make make sure that all zones have the same computing power could be useful to solve this.
  - A different approach would consist on restricting how easily new nodes with high computational power can join or switch zones.
- **At each level, the merging time between forks should be faster than the periodicity of merges**.