Scaling Cryptocurrencies Using Directed Acyclic Graphs to Facilitate Asynchronous Block Creation: A Brief Introduction

Jonathan Healy

December 13, 2018

Abstract

The main purpose of this paper is to briefly examine two asynchronous blockchain consensus protocols; SPECTRE and Hashgraph. Both of these protocols employ the concept of directed acyclic graphs to allow for concurrent block creation. As the block creation rate of a traditional blockchain is sped up, more forks begin to naturally occur and observing this has lead some researchers to conclude that organizing the resulting directed acyclic graph is the best way to address the scaling problem that faces Bitcoin and other similar cryptocurrencies. A main challenge to using this approach is that blocks created concurrently, at close to the same time, may have identical or conflicting transactions that need to be somehow settled and put into linear order. As an exercise, an alternative solution using parallel blockchains where each blockchain processes a different set transactions based on a Sender's address is hypothesized.

1 Introduction

Over the years, Bitcoin and other permissionless cryptocurrency networks have shown themselves to be incredibly slow and not only at times when they are under an unusually high load. The average time it takes to create a block in the Bitcoin network is 10 minutes, setting the transaction processing speed at a mere 3.3 to 7 transactions per second [1]. Furthermore, Bitcoin can only offer eventual consistency and it is recommended that Users wait for up to 6 blocks for confirmation, to ensure that their transaction has not been put into a forked chain and is instead where it should be, on the main blockchain. Altogether this means that it should take roughly one hour for someone to have enough confidence that the money that was sent them has truly been received. This is all under normal circumstances.

At the end of 2017 and into early 2018, during the height of cryptocurrency mania, the Bitcoin network became almost unusable. Indeed, in early December it was taking up to 16 hours at times to receive one confirmation on a transaction. With more people using the network it also became necessary to include an unusually high transaction fee, as Miners can choose to only process transactions with higher fees. In early December, again, average fees for one transaction rose to a price of almost $4 USD [2]. Consider that one of the main use cases for cryptocurrency, in general, is with micro-transactions.

Numerous ideas have been developed and presented attempting to improve the throughput of the Bitcoin network and other permissionless networks like it. In a permissioned network, where there is control over who can operate nodes and do things like approve transactions or vote for a leader, there are less stringent security concerns. In Bitcoin itself, many potential improvements have either been proposed or implemented. People have sought to increase the potential size of each block in order to increase the number of transactions that can be contained but this approach only adds linear improvement.

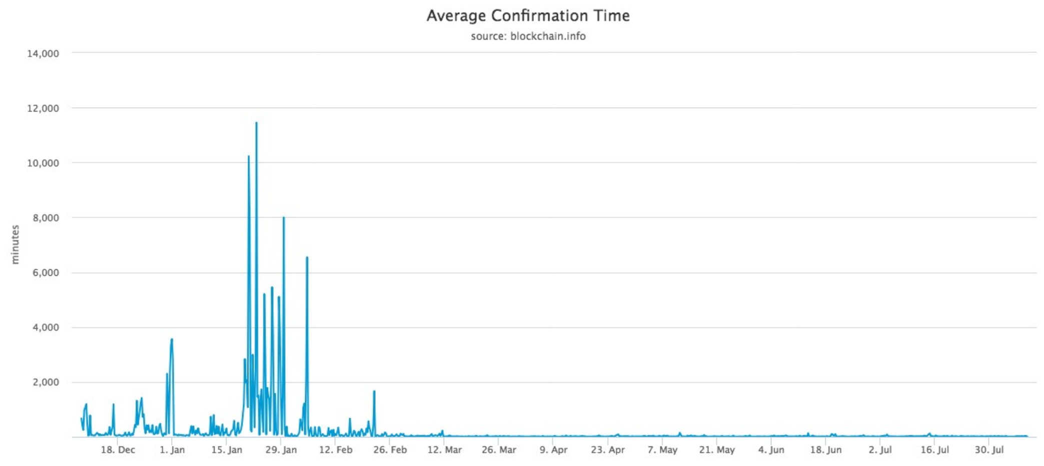


Figure 1. Average Time for One Confirmation in the Bitcoin Network (Dec.17 - Jul.18) [3]

Mining difficulty in Bitcoin eventually increases or decreases in order to theoretically slow down the rate at which new blocks are created and the reason for this is to limit the amount of forking that can happen. Miners sometimes create blocks that will later be pruned off of the main chain. Small block sizes and a low block creation rate helps Bitcoin reach consensus securely. As the block creation rate increases, Bitcoin starts to look more and more like a Directed Acyclic Graph (DAG) because of the increasing number of forks that are naturally created. Observing this has led to a number of DAG based consensus protocols to be either proposed [4] or else implemented as fully online public cryptocurrency networks. Allowing a blockchain to fork by design means that we are accepting concurrency.

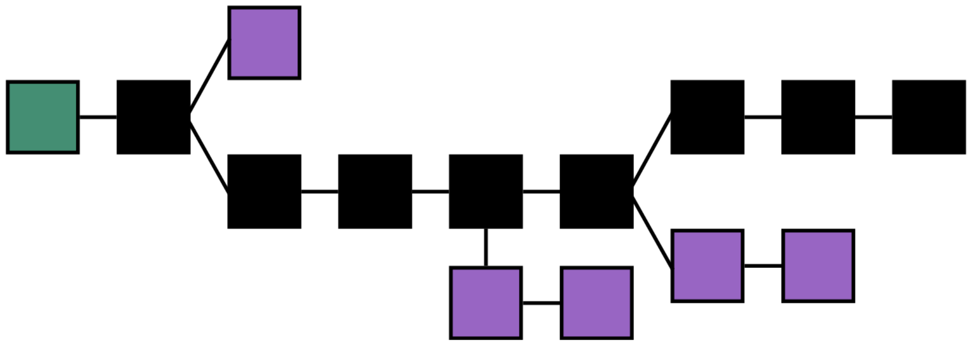


Figure 2. Bitcoin Visualization. The main chain is black. The genesis block is green. Purple blocks are called orphans and are pruned from the main chain. [5]

This paper seeks to examine two of the more popular ideas in the DAG based cryptocurrency space; SPECTRE and Hashgraph. A naive alternative to some of these ideas is also formulated, mostly as an exercise to try to come to an understanding of how an asynchronous blockchain may work.

The first protocol we will look at is SPECTRE (Serialization of Proof-of-Work Events: Confirming Transactions via Recursive Elections) [4]. SPECTRE is a Proof of Work protocol proposed by Yonatan Sompolinsky, Yoad Lewenburg, and Aviv Zohar in 2016. Proof of Work blockchain implementations rely on nodes or miners proving that they have performed a computationally challenging puzzle before others have, giving them the right to add a block of transactions to the main blockchain, receiving a reward for doing so in newly minted cryptocurrency. This differs from other protocols that utilize proof of stake, where a node will be selected to mint a new block of transactions proportionally to the amount of tokens that it is staking on the network. One advantage to proof of work is that it slows down the consensus process, limiting the number of branches or forks that will occur.

Unlike Bitcoin and SPECTRE, Hashgraph is a Proof of Stake consensus protocol. Hashgraph was proposed in 2016 by Dr. Leemon Baird in his paper, "The swirlds hashgraph consensus algorithm: Fair, fast, byzantine fault tolerance." [7] Originally Hashgraph was only implemented in permissioned and private network environments but recently the team behind Hashgraph have released what they call, Hydera Hashgraph, which is their version of a public blockchain [8]. Critics have widely shown some skepticism towards the project, partly due to the fact that the network will be overseen by a governing council of 39 industry leaders [11]. Hashgraph as well is not open source and these are things that do not always sit well in a world where the majority of people want to see almost everything become decentralized. Criticism by and large is not directed at the original Hashgraph consensus algorithm itself and it would be hard for someone to say that the original paper does not at least have some pretty interesting ideas.

1.1 Related Work

IOTA, which is a well-known cryptocurrency aimed at Internet of Things applications utilizes a directed acyclic graph called the Tangle to store transactions [11]. Another publicly available cryptocurrency called Byteball, which was first proposed by Anton Churyumov in 2016 uses a directed acyclic graph for the same reason [12]. An open source version of the Hashgraph algorithm called, OpenHashgraph has been proposed although because Hashgraph itself is not open source this may eventually present some problems [10]. Serguei Popov's initial paper on the Tangle has been cited some 106 times according to Google Scholar. Most of the citations for his paper come from works that do not directly seek to analyze consensus as it pertains to directed acyclic graph, Blockchain-like distributed ledgers. The same authors that published the SPECTRE paper have more recently published a paper called, "PHANTOM, GHOSTDAG: Two Scalable BlockDAG Protocols." [13] and this paper has not been covered for this report.

2 SPECTRE (Serialization of Proof-of-Work Events: Confirming Transactions via Recursive Elections)

SPECTRE aims to enable parallel and concurrent block creation by allowing a directed acyclic graph (DAG) to grow instead of slowing it down and pruning it into a single chain like what is done in traditional blockchain architecture like Bitcoin. SPECTRE is presented as, "a permissionless distributed ledger that establishes high throughput and fast confirmation times while maintaining resilience to 50% attackers." [6] Instead of mimicking Bitcoin and limiting block creation time to one block created every 10 minutes, SPECTRE looks to create several blocks per second.

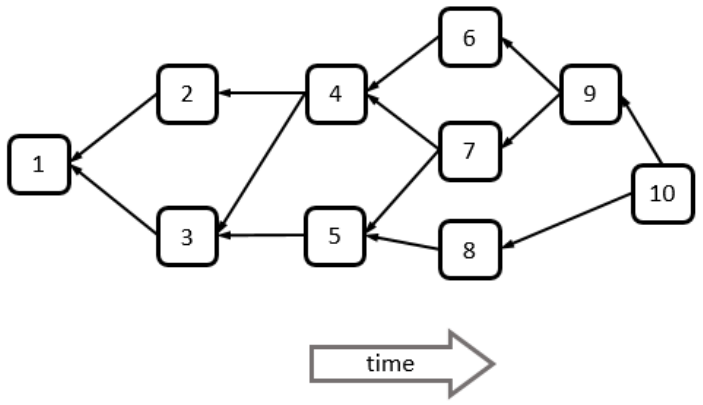
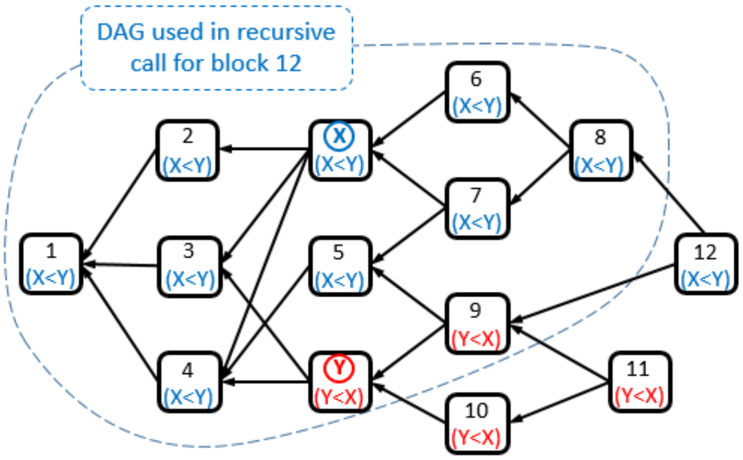


Figure 3. A Block DAG. Each block may refer to several predecessor blocks by including their hash in its header. [6]

Two main properties are guaranteed by SPECTRE; 1. Weak Liveness and 2. Safety. Weak liveness means that transactions issued by honest participants will be confirmed quickly and safety means that once a transaction is approved, it is very unlikely to be double spent or reversed. Because blocks can be created concurrently, transactions will sometimes be duplicated in different blocks. To work around this issue, the SPECTRE protocol creates a set of accepted transactions determined by voting.

Step 1: For each pair of blocks *x*, *y* determine whether "*x* defeats *y*" (denoted *x*<*y*) or "*y* defeats *x*" (denoted *x*>*y*).
Step 2: Output a set of accepted transactions. A transaction embedded in block *x* is considered "accepted" if block *x* defeats all blocks that contain conflicting transactions, and all inputs of *x* were accepted. [6]

Given 3 blocks x, y, and z, z will vote for x if x is in it's past and y is not. If both x and y are in z's past, z will vote for either x or y based on which block is in the majority of it's past. If neither x or y are in z's past, z will vote according to the majority in it's future.



- Blocks *x*, 6-8 all vote *x*<*y* (block *x* is in their past, but *y* isn't).
- Similarly, blocks *y*, 9-11 vote *y*<*x*.
- Blocks 1-5 vote *x*<*y*. This is because they see more *x*<*y* voters in their future than *y*<*x* voters.
- Block 12 votes according to a recursive call on the DAG that does not contain blocks 10,11,12 (in this case the recursive call has each block voting exactly the same, but it is not necessarily true for all DAGs).

Figure 4. Voting in SPECTRE. [6]

At first glance it may seem like there could be a significant amount of computation involved in determining a set of accepted transactions, especially at the high block creation rate that has been hypothesized for SPECTRE but it has been claimed that this really is not too much of an issue [6]. At a slow enough block creation rate, SPECTRE is simply a Bitcoin blockchain and voting becomes equivalent to the outcome that arises from the longest chain rule which is the technique used to prune orphan blocks in the Bitcoin protocol. Like Bitcoin, SPECTRE employs the concept of difficulty in order to limit the number of blocks created per second although obviously SPECTRE is aiming to be a much faster network with many more transactions processed per second.

With parsing blocks to create sets of accepted transactions, a real concern is that allowing blocks to be created in parallel will furthermore lead to separate blocks having identical transactions. The creators of SPECTRE claim that miners will be incentivized to randomly select transactions from their mempool, therefore reducing the number of collisions. This is especially applicable if the mempool is large and there are a significant number of Users trying to send transactions through the network. If SPECTRE is theoretically being under used however, as many cryptocurrency networks are, one might wonder what advantages it has over the Bitcoin protocol. If a protocol like SPECTRE can scale with demand and still offer similar security to what is offered by traditional proof of work blockchains in a permissionless setting then SPECTRE could find itself becoming quite popular especially as cryptocurrency in general seeks to evolve and eventually disrupt centralized payment providers like Visa and PayPal.

3 Hashgraph

Hashgraph, also called the *Swirlds hashgraph consensus algorithm*, was created by Dr. Leemon Baird as a consensus algorithm for replicated state machines [7]. Hashgraph claims to guarantee Byzantine fault tolerance through virtual voting. By using a gossip protocol and gossiping about gossiping, events build a shared history called a Hashgraph that ensures fairness as it should be almost impossible for any one actor to try to tamper with the ordering of transactions. Events are similar to blocks in the Bitcoin blockchain except that an event contains two extra hashes, one pointing to the event that sent a process it's latest message and the other pointing to it's last event. The idea seems similar to using vector clocks to achieve partial ordering except that more information is passed between events and events start to form a total picture about the ordering of events in the network.

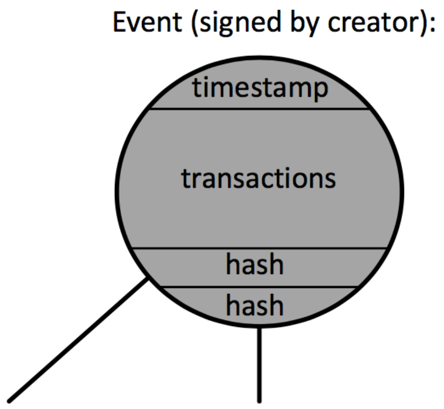


Figure 5. An Event in Hashgraph. [10]

Unlike Bitcoin and SPECTRE, Hashgraph is meant to be employed as a proof of stake algorithm and claims to offer asynchronous Byzantine Fault Tolerance (aBFT). aBFT is apparently the strongest form of Byzantine Fault Tolerance and can offer consensus even if up to 1/3 of the network is malicious and can act to slow down or delete messages sent between participants. The Bitcoin protocol on the other hand does not guarantee Byzantine tolerance because an agreement on the state of the ledger can never be guaranteed with only the probability of such a state rising over time as more blocks are added. Also, Bitcoin does not deal with network partitioning immediately as isolated miners may build separate chains that do not agree on an ordering of transactions.

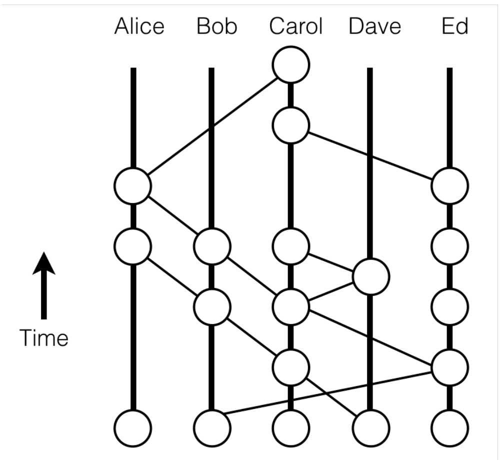


Figure 6. Gossip history as a directed graph representing the history of the gossip protocol as a directed acyclic graph. "When Alice receives gossip from Bob, telling her everything he knows, that gossip event is represented by a vertex in the Alice column, with two edges going downward to the immediately preceding gossip events by Alice and Bob." [7]

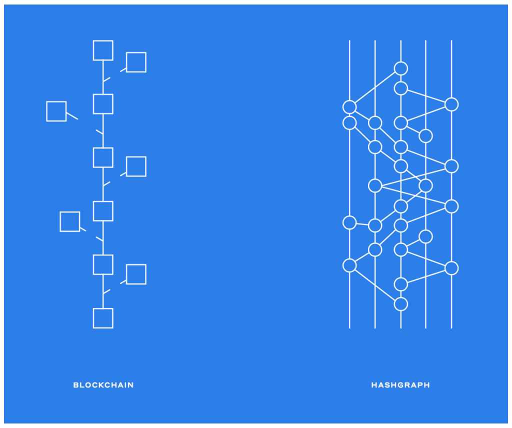


Figure 7. Conceptually, a blockchain is like a tree that is constantly pruned while in a hashgraph, the tree is allowed to grow and the new branches are woven back into the body. From the Hedera Hashgraph white paper. [8]

In addition to achieving partial ordering through the use of it's gossip about gossip protocol, Hashgraph gives transactions what is called a consensus timestamp which represents an average of when other nodes in the system first received news about a transaction. This is meant to help the system achieve fairness as a malicious node would not be able to influence the value of this consensus timestamp by very much. Transactions are ordered according to their consensus timestamp. By using gossip, Hashgraph eliminates the need to pass messages to vote on decisions which could theoretically take O(n^2) or even O(n^3) messages to come to an agreement over every single decision. Voting in Hashgraph is virtual as nodes in the network already know how other nodes will vote based on sending random messages to each other sharing news about the shape of the Hashgraph.

> As the hashgraph grows upward, the different members may have slightly different subsets of the new events near the top, but they will quickly converge to having exactly the same events lower down in the hashgraph. Furthermore, if Alice and Bob happen to both have a given event, then they are guaranteed to also both have all its ancestors. And they will agree on all the edges in the subgraph of those ancestors. All of this allows powerful algorithms to run locally, including for Byzantine fault tolerance. [7]

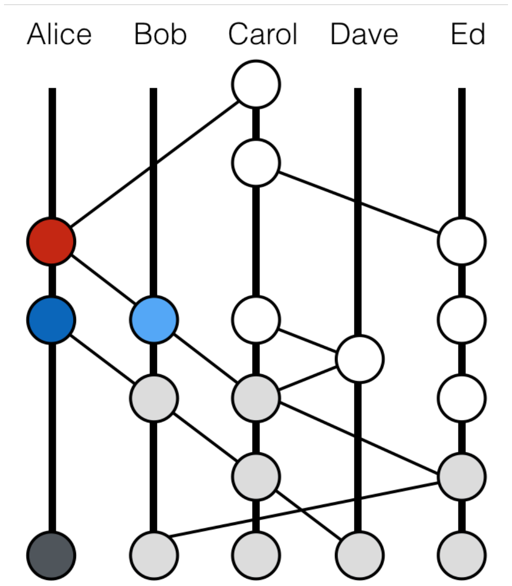


Figure 8. The hashgraph data structure. The red node represents an event created by Alice recording the occurrence of Bob sending her a message, telling her everything he knows. "The event contains a hash of two parent events (blue): the self-parent (dark blue) by the same creator Alice, and the other-parent (light blue) by Bob. It also contains a payload of any new transactions that Alice chooses to create at that moment, and a digital signature by Alice. Other than those two, the ancestor events (gray) are not stored in the red event, but they are determined by all the hashes. The self-ancestors (dark gray) are those reachable by sequences of self-parent links, and the others (light gray) are not." [7]

The Hashgraph algorithm uses a concept called, 'strongly seeing' to come to consensus about a linear ordering of events. In a network with n nodes, an event w is said to strongly see an event x if w can see 2n/3 events by different nodes who can all see x. This allows Hashgraph to guarantee Byzantine fault tolerance through the use of virtual voting.

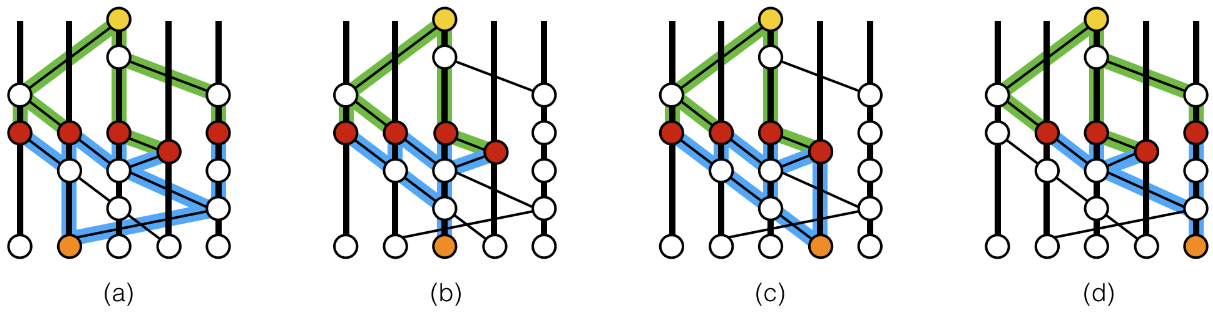


Figure 9: "Illustration of strongly seeing. In each hashgraph, the yellow event at the top can *strongly see* one of the orange events on the bottom row. There are n = 5 members, so the least integer greater than 2n/3 is 4. In (d), one event (orange) is an ancestor of each of 4 intermediate events by different creators (red), each of which is an ancestor of the yellow event. Therefore, the yellow event can strongly see the orange event. Each of the other hash-graphs is colored to show the same for a different orange event on the bottom row, which the yellow event see through at least 4 red events." [7]

Hashgraph relies heavily on the notion of 'fairness' to explain itself and claims that while distributed database protocols like Paxos have Byzantine tolerance they don't ensure fairness to all participants. Because consensus timestamps are calculated via fair voting between all nodes, Hashgraph is a leaderless system and this helps eliminate a central point of failure. Hashgraph introduces three concepts of fairness:

1. Fair Access: The random nature of the gossip protocol prevents malicious nodes from trying to prevent other nodes from hearing about a transaction.
2. Fair Timestamps: Calculating the median timestamps for transactions in what is called a consensus timestamp ensures that malicious actors cannot overly influence the ordering of events.
3. Fair Transaction Order: This concept stems from having fair timestamps and is important in any type of setting that calls for the application of a distributed ledger.

Many observers agree that the Hashgraph consensus protocol has merit, however as a permissioned network it is not competing on the same playing field against traditional permissionless Blockchain environments like Bitcoin. Many other permissioned protocols can offer high throughput as well and it is not entirely clear on how Hashgraph improves on other consensus algorithms in that space. Decentralization is a key word for the cryptocurrency community and many people had hoped that the team behind Hashgraph would have the ability to offer the world a better alternative to traditional blockchains. By and large, blockchains are either decentralized and slow or else faster but only when accompanied by a more centralized architecture. Hedera Hashgraph, Leemon Baird's public distributed ledger, for whatever reasons, did not deliver anything new for fans of decentralization. Potentially however, even though Hashgraph is not open source, there are some interesting ideas that could maybe inspire a more robust and permissionless architecture in the future.

3 Alternative Implementation

A very simple asynchronous Blockchain implementation has been provided with this report as part of the final project for CSC 464: Concurrency, offered at the University of Victoria. As determining a linear order of events for transactions is of utmost importance in any of these systems it is being hypothesized here that a simpler alternative could be to run parallel Bitcoin-like proof of work chains that naturally fork by dividing the transactions they are allowed to handle by the transaction Sender's blockchain address. By using a system like this there should definitely be fewer transactions to order as only accounts that are sending value need to be securely accounted for initially. This partly relates to the possibility of mounting a double spending attack on the network. By assigning random address spaces to miners and hashing power, something like this might make it harder for an enemy with a concentrated amount of computing power to launch conflicting transactions for financial benefit.

Theoretically in this system, there would be a main chain and at certain time intervals parallel chains would settle on the main chain incrementing and decrementing all of the accounts that have sent and received transactions. Any money sent in the previous rounds would need to settle on the main chain before a receiving address could spend that same money. This would be done to ensure that parallel chains do not have to worry about conflicting with each other. For instance, transactions sent by Alice would only be handled on the A chain. Bob could spend his money that was being accounted for on the B chain but if Bob ran out of money he would have to wait for a round to end and settlement to occur before he could spend money he had maybe received from Alice or others in the meantime.

After settling, parallel chains would be created again using the latest block in the main chain as their genesis block with the number of chains for that round being determined by demand based on the amount of recent network traffic. The main chain itself would be almost equivalent to a traditional Bitcoin-like blockchain and would act as a distributed ledger accounting for the entire history of transactions. Nakamoto consensus, which is what is used in Bitcoin has proven itself to work relatively well and replicating the logic on numerous chains is an approach that might have some merit. Numerous people and projects have been struggling with the challenge of increasing throughput in distributed ledgers while keeping a network public, open, decentralized, and permissionless.

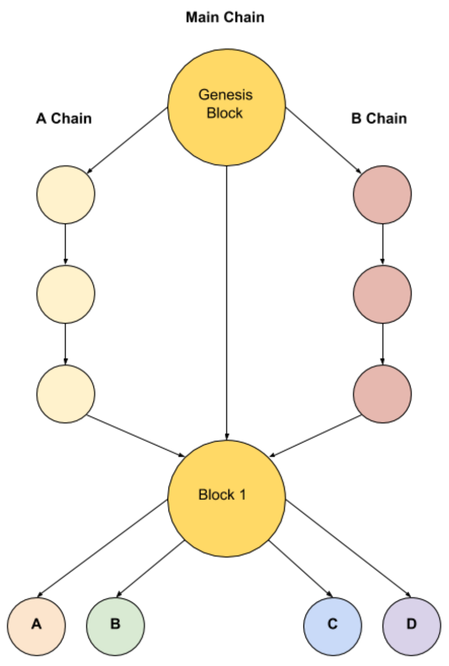


Figure 10: Parallel chains, A and B, fork off from the Genesis Block on the Main Chain. The two chains divide the address space in half and only process transactions that are sent from accounts that belong to either one half or the other. Later, they append their processed transactions, on the main chain, creating Block 1. At this point, network traffic has increased substantially and four chains are deployed, dividing the address space in four.

4 Conclusion

This report has attempted to briefly analyze two consensus protocols, SPECTRE and Hashgraph, both of which use directed acyclic graphs to organize blocks allowing for asynchronous block creation and higher throughput than traditional Bitcoin like Blockchains. Both systems have a lot of merit on the surface and provide interesting solutions to the issue of reaching consensus and ordering transactions by processes operating in parallel, appending a distributed ledger sometimes with conflicting or duplicate information. Any protocol in this space needs thorough real world testing as the value of money that could potentially be at stake on many distributed ledgers makes finding insecurities very attractive to a whole host of bad actors. For almost 11 years, the Bitcoin network has proven itself in the field and as of today there is almost $1.9 billion USD in value stored on it's ledger. The naive attempt at a solution presented in this report simply seeks to replicate Nakomoto consensus on parallel chains that are divided by address spaces. Beyond a simple implementation coded for a final project and a few paragraphs explaining how it may work, there is no other substance to this proposal although it may be interesting to do more work on it in the future.

References

[1] "Bitcoin Scalability Problem." *Wikipedia*, Wikimedia Foundation, 9 Dec. 2018, en.wikipedia.org/wiki/Bitcoin_scalability_problem.

[2] "How Long Do Bitcoin Transactions Take?" *CoinCentral*, 9 Aug. 2018, coincentral.com/how-long-do-bitcoin-transfers-take/.

[3] "Average Confirmation Time." *Blockchain.com,* www.blockchain.com/charts/avg-confirmation-time?timespan=1year.

[4] Yonatan Sompolinsky, Yoad Lewenburg, and Aviv Zohar. "SPECTRE: A Fast and Scalable Cryptocurrency Protocol." *IACR Cryptology ePrint Archive 2016* (2016): 1159.

[5] "An overview of Proof of Work based blockchain consensus protocols (part 1)." *Medium,* 22 Mar. 2018, https://medium.com/@drstone/an-overview-of-proof-of-work-based-blockchain-consensus-protocols-part-1-e04102885093

[6] Zohar, Aviv. "SPECTRE – Aviv Zohar – Medium." *Medium.com*, Medium, 18 Dec. 2016, medium.com/@avivzohar/the-spectre-protocol-7dbbebb707b5.

[7] Baird, Leemon. "The swirlds hashgraph consensus algorithm: Fair, fast, byzantine fault tolerance." *Swirlds, Inc. Technical Report SWIRLDS-TR-2016* 1 (2016).

[8] Baird, Leemon, Mance Harmon, and Paul Madsen. "Hedera: A Governing Council & Public Hashgraph Network." (2018).

[9] Choe, Myongsu. "OPEN HASHGRAPH: AN ULTIMATE BLOCKCHAIN ENGINE."

[10] Baird, L. "Hashgraph consensus: Detailed examples." *Swirlds Tech Report. Swirlds* (2016).

[11] Blockonomi. (2018). *What is Hedera Hashgraph Consensus & How Does it Work?*. [online] Available at: https://blockonomi.com/hedera-hashgraph-consensus/ [Accessed 12 Dec. 2018].

[12] Popov, Serguei. "The Tangle." (2018)

[13] Churyumov, Anton. "Byteball: A decentralized system for storage and transfer of value." *URL https://byteball. org/Byteball. pdf* (2016).

[14] Sompolinsky, Yonatan, and Aviv Zohar. "PHANTOM, GHOSTDAG."