BigTangle Technical Whitepaper

Maximilian Hensel, Kai Cui,
Maximilian Lowin, Jianjun Cui

April 15, 2018

Abstract

Recent cryptocurrencies have tackled a number of key issues preventing widespread adoption and practical usage, one of which is the scalability issue. In this technical paper, a novel big data approach to scalable cryptocurrencies employing state-of-the-art big data technologies such as Apache Spark and Apache Kafka in combination with the highly scalable Tangle DAG base architecture is proposed, thereby achieving high performance in terms of throughput by parallelized computations on a general directed acyclic block graph instead of a blockchain. This feeless and scalable Proof-of-Work approach with mining is projected to solve issues of other recent financial cryptocurrency solutions by providing inter alia native support for token issuances, resulting in attractive use-cases for banks, stock exchanges and companies.

I. Introduction

Since the inception of Bitcoin in 2009 [4], new cryptocurrencies have been rising in popularity and adoption rate and are predicted to potentially rise to rival fiat currencies in the future. Transitioning to such a digital currency offers various advantages, of which the authors believe the infrastructure cost savings in the financial sector via revolutionizing highly complex and opaque traditional transaction systems from current banking and stock market technologies to be the most attractive. To attain this attractivity, two key properties are required: scalability and the absence of fees in general. Additionally, low transaction confirmation times, ease of use, security and anonymity are other baseline properties required for widespread adoption. Traditional Proof-of-Work-based blockchain approaches fail to meet these requirements, while Proof-of-Stake-based approaches face other problems such as politics with regards to mining, voter apathy etc.

In this paper we propose BigTangle, a cryptocurrency network seeking to pioneer an alternative scalable Proof-of-Work solution to recent Proof-of-Stake-based blockchains. The proposed solution's architecture extends the Tangle [1], a DAG generalization of blockchains capable of scaling to an unlimited amount of transactions per second and no transaction fees without compromising other aforementioned properties. Beyond the decentralization of payment processing, the implementation can be used as a base service for the decentralization of markets, the transfer and ownership management or authenticity proofs of any assets. It supports all functions supported by Bitcoin, including but not limited to escrow transactions, bonded contracts, third-party arbitration, multiparty signatures etc.

In contrast to other recent high performance cryptocurrencies, BigTangle employs a Proof-of-Work-based mining process where in principle every transaction is rewarded and helps secure the network. This mining process incentivises users to run full nodes and secure the network, providing advantages such as inflationary currency models and value through economic coupling to the real world. The full node operators and miners are compensated with a time-normalized amount of new BigTangle tokens proportional to their contribution to network validation.

BigTangle starts off as a completely decentralized solution which will not rely on a centralized coordinator to finalize consensus and sees itself as a successor to Bitcoin focusing on other key use-cases than machine-to-machine and differing in a number of key design points as mentioned further below. Projected practical use cases include payment processing, decentralized stock exchanges, supply chain and product integrity tracking.

In the authors' opinion, a transaction system capable of supporting the global transaction volume of banks and stock markets requires computer clusters to be able to work as full nodes. BigTangle will therefore employ established industrial grade big data technologies such as Apache Kafka, Apache Spark and its GraphX API etc. By utilizing local approximations to the non-scalable graph computations of the asynchronous Tangle as described further below in addition to big data computation technologies, it is believed that the proposed solution suffices the properties demanded above. Minimal end user client requirements are a side effect, allowing end users to participate and issue transactions without having to store and handle huge amounts of data but instead providing hashing power and therefore mining rewards only.

II. Overview

The protocol maintains a public ledger of transactions which are contained within blocks. In difference to blockchain-based solutions, blocks reference two previous blocks to form a directed acyclic graph (DAG) instead of a chain. Per protocol, a block is valid if the block itself is formally correct, all indirectly referenced blocks are valid and no conflicts, e. g. double spends, are approved. Contributing blocks when issuing transactions to the network therefore helps securing it by intending to approve the referenced blocks after successful validation. A more detailed explanation of the base architecture and existing concepts can be found in [1].

Participants are connected in a standard peer-to-peer network via a gossip protocol as fall-back solution. In addition, BigTangle will employ Apache Kafka data streaming to achieve sufficient scalability and propagation speed. The network is split into two different archetypes: clients and server network nodes. Generic end users and miners can participate as clients that issue transactions with the aid of a network node, using their hashing power to create blocks and solve the low-difficulty Proof-of-Work themselves, while network nodes maintain a (potentially pruned) copy of the graph and provide validated pairs of blocks to approve that are conflict-free.

The network nodes can derive account balances and transaction states from the graph and provide the maintained states to clients in exchange for hashing power. Furthermore, a network node operator might choose to also participate in the mining process himself. To do so, he will simply use his computational power to solve Proof-of-Work for new empty blocks and thereby validate their respective parent blocks.

The server-side validation as found in [1] can be approximately computed effectively by utilizing local confirmed state approximations. The network node creates their local view of the tangle by maintaining helper constructs such as local confirmed state snapshot and their UTXO state, and the validation is performed with help of this snapshot. More precisely, MCMC selects the unconfirmed block pair and the validation is performed on the set of blocks to be approved in addition to the snapshot, resulting in other blocks considered confirmed but not approved to also be included in the validation. Further information to this process can be found below.

Baseline properties provided by BigTangle are high hash power security due to the mining process, infinite scalability, sufficiently fast transaction confirmation times, full decentralization, trustlessness and permissionlessness, feeless transactions, in-principle quantum security and normalized inflation. To make use of these properties, BigTangle will natively support custom token issuances and decentralized token exchanging.

Popularly quoted future solutions to scalable cryptocurrencies include highly complex sharding, voting-based and capital-based Proof-of-Stake or workarounds such as the lightning network [3] with the potential for exorbitant fees. Instead, we propose a permissionless solution that returns to the well-known Proof of Work approach of the original Bitcoin solution. Instead of fees, mining serves as a network maintenance incentive. Since it is impossible to allow every device to participate equally as network nodes while at the same time achieving infinite scalability without employing highly complex sharding technologies, it is intended that network nodes are mostly deployed in sufficiently big computer clusters utilizing state-of-the-art big data technology. Instead, clients simply cooperate with network nodes by providing mining revenue and building blocks as needed.

Nonetheless, the network stays permission-less and avoids all central instances such as coordinators. In principle, as long as the nodes fulfill the minimum requirements to keep up with the transaction volume, anybody can host a full node and participate in the network and it is intended for many full nodes to exist regardless of mining rewards. For example, super market chains or banks can deploy their own full nodes to process their transaction volume without being dependent on other full nodes. By utilizing Proof-of-Work, we can avoid the typical pitfalls of Proof-of-Stake-based inflation models.

III. Technical Details

In the following, we briefly discuss key technical advances and their realization. For visualizations of established concepts, please refer to existing literature.

i. Implementation

To achieve high scalability, the node implementation is built upon industry standard big data technologies, including but not limited to Apache Hadoop, HBase, Kafka and Spark. The codebase used for initial development is the well-known Java Bitcoin implementation called bitcoinj.

ii. Cryptographic Components

BigTangle currently relies on elliptic curve cryptography for multi-use signatures. The curve used is Bitcoin's Secp256k1. Similarly, the public key hashes are also Base-58 encoded, allowing Bitcoin users to reuse their addresses in BigTangle. The Proof-of-Work hashing function implemented is the well-known ASIC-resistant CryptoNight algorithm. Due to the low difficulty parameters used in the Proof-of-Work of BigTangle, the base architecture is quantum resistant in principle as shown in [1] except for the signature scheme. In case of breakthroughs in quantum computing, support for new signature schemes can easily be added.

iii. Transactions

Transactions are based off of Bitcoin's Unspent Transaction Output (UTXO) model. Users can issue valid transactions as long as they can provide a valid input script for the UTXOs referenced in the transaction. The UTXOs use Bitcoins not Turing-complete stack language. thereby allowing for the same set of functionality as found in Bitcoin extensions. We briefly list all three basic transaction types:

Default Transactions are typical transactions with support for different tokens.

Coinbase Transactions are used in the periodic mining reward blocks to reward the Proof-of-Work put into validating blocks.

Token Issuance Transactions are used in token issuance blocks to issue custom tokens. They are legitimated by the token's corresponding private key signatures.

iv. Corresponding Blocks

A block consists of its header and multiple transactions. In addition to Bitcoin, the header contains an additional reference to an older block, a miner address, a type field for the types as seen below and additional data for the types listed below. We briefly list all three basic block types:

Default Blocks contain any of transactions. The majority of mining blocks are projected to be empty when miners have not been delegated real transactions to perform.

Coinbase Blocks only consist of the block header and in serialized form do not contain transactions. They are used in the mining process as detailed further below and their coinbase transactions are computed on confirmation.

Token Issuance Blocks contain at least one token issuance transactions of a specific token. On the first issuance of a token only, we can specify if repeated token issuance blocks of this token are allowed or disallowed.

v. Participants

Participants can take on different roles in the network depending on their available computational power and bandwidth. The different possible types of participation are ordered in descending requirements of bandwidth, space and computational power. We briefly list all three basic participant types:

Full Nodes keep a copy of the full tangle saved locally. They can fully participate in the network and provide any requested blocks on request.

Pruning Nodes maintain a pruned version of the tangle. Only the most recent blocks in terms of confirmation are kept in storage. The node can still fully participate in the mining process unless a highly unlikely reorganization event happens.

Clients do not keep a copy of the Tangle. They rely on the nodes to provide them with necessary information to create transactions and blocks. Clients can solve the proof of work of their issued transactions and blocks to incentivize nodes to broadcast them for mining revenue.

vi. Protocol Details

Node Maintenance
In the following, we describe assorted technical details required for preparing the Tangle base architecture to achieve elegant scalable operation. For further more detailed information, please read the yellow paper and code of BigTangle on release.

As mentioned before, the BigTangle node implementation locally maintains the milestone, a set of blocks it considers as confirmed and thereby in principle finalized. In addition, we maintain additional auxiliary information and block evaluations such as rating, cumulative weight, depth, height etc. as found in [1]. The milestone update process in simplified form consists of the following steps:

1. Ingest new blocks such that a valid Tangle is obtained.
2. Update relevant evaluations only.
3. If needed, remove now unconfirmed blocks and dependants from milestone.
4. Find new locally confirmed blocks, resolve conflicts and add to milestone.

We consider locally confirmed any block that has reached the upper confirmation threshold in terms of rating and is sufficiently deep. We also add a hysteresis to removing blocks to prevent unnecessary reorganizations.

Regarding the setting of the confirmation threshold parameters for removal from and addition to the milestone, we assume that at least $66.\overline{6}\%$ of the blocks are honest and validate correctly. To ensure that the honest miners always find a consensus, we set the lower confirmation threshold to $66.\overline{6}\%$ as shown further below. For stability purposes mentioned above, we then set the upper confirmation threshold to 75%

Of particular note is the **conflict resolution algorithm**. In principle, this conflict resolution algorithm should only find application in step 4 during a successful attack, since an honest network would only approve conflict-free block combinations. Nonetheless, this case must be handled correctly by the network in case of attacks and is also used during the tip selection mentioned below. In short, we simply include conflicts in descending order of maximum occurring rating, taking great care to eliminate losing candidates and all their approvers and dependants from the lists of other conflicts and blocks.

Lastly, we may prune no longer relevant blocks and their evaluations to prevent the Tangle from growing indefinitely in terms of storage space, instead keeping only their block hash. For example, blocks in the milestone and their conflicting counterparts if any can be pruned after reaching sufficient depth, age of confirmation, inverse height etc.

Validation and Approval Selection

When generating a new block, we require two previous blocks to reference such that no conflict is created. To find these two blocks, we first apply an MCMC algorithm similar to the approach shown in [2] to find single tips. Then we resolve conflicts as detailed before and reverse on the path taken until a resolution point is found. We repeat the conflict resolution procedure for the pair of blocks and obtain two blocks that are conflict free.

Note that the resolution and validation is performed on the current milestone state of the node and therefore approximating the validation algorithm detailed before by adding more validation constraints. A complete list of added validation points is omitted at this point.

Mining Process

To incentivise node operation and network validation, we introduce a mining process with similarities to the Bitcoin mining process. Under the assumption that the Tangle's relevant tips generally are of approximately equal height, we propose the following reward scheme:

At least one reward height interval higher than the reward height interval to be rewarded, coinbase blocks are issued which include a field to identify their reward height interval. All referenced blocks in this interval are rewarded by it and it therefore must be in conflict with coinbase blocks of the same reward height interval.

Since the Tangle is an asynchronous network, we cannot simply reward what is seen locally, as every node sees a different version of the Tangle. Instead, we must introduce a fix point such that every node can calculate the same rewards. The coinbase block itself will be this fix point. Using only the blocks approved by the coinbase block itself, we can compute consistent rewards since we know the Subtangle to be unbroken in order for the coinbase block to be considered for confirmation. In addition, the calculation of rewards is done locally and on demand only, e. g. when entering the milestone.

The key problem to solve here is how to only approve coinbase blocks that are consistent and fair according to the nodes local Tangle state. The solution is to use the following validity constraints for coinbase blocks: Most of their referenced blocks of the specified height interval must be locally confirmed and most of the locally confirmed blocks of the specified height must be referenced at the time of reception. To prevent deadlocks due to local inconsistencies, we must also allow this constraint to be overridden by sufficient rating and depth, as in that case the rest of the network has accepted the block as valid.

Additionally, we must include countermeasures against non-validating miners' attacks as shown further below. For example, we can punish the blocks of each height with the lowest cumulative weight as seen from the fix point, since non-validating miners' will pursue a suboptimal tip selection algorithm to avoid validating new blocks and therefore gain less cumulative weight.

Now it becomes apparent why we used the fix point and empty transaction list approach: Since we need to punish non-validating miners' blocks, it is easier to detect local consistency of referenced blocks at reception than it is to evaluate the inconsistency level of cumulative weight from a fix point view. For Denial-of-Service prevention purposes, we also increase the difficulty targets accordingly.

A per-transaction reward for the next interval can be calculated analogously to Bitcoin's difficulty adjustment system: By enforcing a monotonous increase of timestamps of blocks and limiting the validity of timestamps from the future, the per-transaction reward is adjusted according to the current transaction rate to enforce an approximately constant coin emission rate of e. g. 2% of total supply p.a. without the risk of manipulation.

Token Issuance Process

The token issuance process is trivial: We can simply legitimize a new token issuance block by signing their transactions with the token's corresponding private key. A token is identified by its ID in form of a public key hash. The first time issuance configuration either allows or disallows repeated issuances.

IV. Attack Schemes

We assume that at least $66.\overline{6}\%$ of the blocks are honest and validate correctly due to the game-theoretic mining revenue penalties applied to non-validating miners. For up to $33.\overline{3}\%$ of malicious hashing power, we assume that although hashing power does not directly correlate with rating influencing, only up to $33.\overline{3}\%$ of the rating is hijacked by the attackers for relevant blocks. It is likely that the percentage of hijacked rating tips are significantly lower for most relevant blocks, and further mitigation can be provided by including older snapshots into the rating calculation algorithm. As such, we must ensure that the honest miners still find a consensus:

- In case $x\%$ of malicious rating tips approve double spends, we must ensure that no reorganization occurs, meaning that the lower threshold must be below $(100 - x)\%$.

- In case $x\%$ of malicious rating tips approve a conflict, we must ensure that no network split occurs, meaning that the lower confirmation threshold must be above $(\frac{100-x}{2} + x)\%$ to prevent one half of the honest hash power seeing a different version than the other half.

It is obvious that the maximum possible x for which such a lower confirmation threshold exists is $x = 33.\overline{3}$. The corresponding lower confirmation threshold therefore is $66.\overline{6}\%$. For stability purposes mentioned above, we then set the upper confirmation threshold to 75%

Other attacks include parallelizing Proof-of-Work to accumulate the highest cumulative weight, which is mitigated by the probabilistic nature of MCMC as well as network latency and milestone update rate in general being slower than Proof-of-Work computations.

Not validating any transactions runs the risk of wasting great amounts of hashing power, while building your own subtangle by approving your own blocks only will lead to lower cumulative weights compared to other blocks of the same height and a corresponding economic punishment plus orphaning risk.

Trying to relink your prebuilt subtangle of higher height to circumvent gaining less cumulative weight is mitigated by penalizing high height difference transition probabilities. This will also ensure a smooth height distribution for relevant tips.

For other attack vectors such as double spends and their game-theoretic effect, please refer to [2] and [1].

V. Practical Use Cases

As mentioned before, projected practical use cases include the substitution of various currently costly and trust-based technical processes. In the following, the use cases are briefly explored.

i. Payment

An obvious main use case is payment processing. By providing a scalable infrastructure, BigTangle enables the global transaction volume to be processed in one network. Most importantly, this offers infrastructural cost advantages by eliminating complex and costly processes of traditional payment processing for banks, companies and the general populace. Note that the network hashing power is approximately proportional to the BigTangle internal token and is therefore decoupled from concrete usage, theoretically resulting in downwards unbounded upkeep at the cost of increased confirmation times compared to traditional static infrastructure costs. At the same time, adequate processing speeds scaling logarithmically with the transactions per second can be achieved.

ii. Fiat Money

For banks, the token issuance protocol can be used to issue bank-backed tokens denoting conventional fiat money. Since the issuance in principle requires little to no participation in keeping up the network, BigTangle is a low cost solution for all parties. Transactions of fiat money can then feasibly be processed within seconds on a global scale.

iii. Stock Markets

Markets for stocks, bonds etc. can easily be realized by creating new token equivalents. Companies can publish stocks and use the BigTangle network, essentially substituting costly stock exchange processes by the feeless BigTangle processing network.

Examples for the largest segments that will be affected: Bonds, Swaps, Derivatives, Commodities, Unregistered/Registered securities, Over-the-counter markets, Collateral management, Syndicated loans, Warehouse receipts, Repurchase markets etc.

iv. Micro Transactions

Service fees can now be charged in microcent values or via seconds of hashing power due to the departure from winner-takes-it-all, allowing for new business models, e. g. online newspapers.

v. Supply Chain Management

Assuming trustworthy producers that issue authenticity tokens, it is trivial to track product authenticity via token transfers. This use case extends into classic supply chain management.

VI. Future Roadmap

We briefly list potential future extensions to BigTangle.

i. Privacy

There are a multitude of arguments for if private transactions should exist. Nonetheless, it is an interesting extension to the BigTangle network and should be investigated.

ii. Smart Contracts

Although in the eyes of the authors Turing-complete smart contracts are not required for attaining the main goal of infrastructure cost savings, general smart contracts for BigTangle will be subject of future investigations.

References

[1] Popov, S. (2016). The tangle. https://iota.org/IOTA_Whitepaper.pdf

[2] Popov, S., Saa, O., & Finardi, P. (2017). Equilibria in the Tangle. arXiv preprint arXiv:1712.05385.

[3] Poon, J., & Dryja, T. (2016). The bitcoin lightning network: Scalable off-chain instant payments. draft version 0.5, 9, 14.

[4] Nakamoto, S. (2008). Bitcoin: A peer-to-peer electronic cash system.

[5] Wood, G. (2014). Ethereum: A secure decentralised generalised transaction ledger. Ethereum Project Yellow Paper, 151, 1-32.