BigTangle Whitepaper

Kai Cui, Jianjun Cui,
Maximilian Hensel, Maximilian Lowin
kai.cui@bigtangle.net

July 29, 2018. Version 1.5

Abstract

BigTangle is a cryptocurrency network extending directed acyclic graph architectures with Markov Chain Monte Carlo (MCMC) as consensus algorithm. Through the use of industry-grade big data technology in conjunction with its parallelizable architecture, BigTangle is a successor to Bitcoin that can fulfill economically important key use-cases. Proof-of-Work mining, smart contracts and custom token self-issuances are supported. Key Features: Proof-of-Work Mining, Completely Feeless, Real-Time Confirmation, Infinite Scalability, Permissionless, Trustless, Decentralized, Distributed Proof of Work and Quantum Security.

I. Introduction

Since the inception of Bitcoin in 2009 [1], new cryptocurrencies have been rising in popularity and could 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 from revolutionizing highly complex and opaque traditional transaction systems of current banking and stock market technologies to be the most attractive.

To achieve 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 [7], while Proof-of-Stake-based approaches [20] might face other problems such as voter politics, mining regulation etc.

In this paper we propose BigTangle, a scalable Proof-of-Work-based cryptocurrency as a protocol for the internet of value. The proposed solution's architecture is a directed acyclic graph (DAG) generalization of blockchains with MCMC as consensus algorithm [4] that is capable of scaling to infinite transactions per second while remaining feeless. The recently popularized DAG architectures [4][8][9] have the potential to scale beyond traditional blockchains and avoid problems of blockchains such as block forks [15] and scalability issues [7].

BigTangle's features are a superset of features supported by Bitcoin, including but not limited to escrow transactions, bonded contracts, third-party arbitration, multiparty signatures and its stack language Script in general. The Bitcoin blockchain is a special case in BigTangle's DAG architecture. Using the mining reward process detailed later, it is possible to change the BigTangle's parameters to return to a conventional blockchain.

In contrast to most recent high performance cryptocurrencies, BigTangle employs a Proof-of-Work-based mining process where most honest transactions are rewarded and help securing the network. This mining process incentivises users to run full nodes and secure the network, providing the advantages of inflationary currency models and value through economic coupling to the real world. Full node operators and miners are therefore compensated with a time-normalized amount of new BigTangle tokens proportional to their mining contribution.

BigTangle is a successor to Bitcoin that focuses on a variety of economically important key use-cases: Beyond the decentralization of payment processing, the network can be used as a base service layer for the decentralization of markets in general, transfer and ownership management, authenticity proofs for assets of any kind or supply chains and ownership management as it has been described in as early as 1998 [25].

In the authors' opinion, a transaction system capable of supporting the global transaction volume of banks and stock markets requires sufficiently performant computer clusters to work as full nodes. BigTangle employs established industrial grade big data technologies such as Apache Kafka, Spark and its GraphX API.

By utilizing local approximations to previously non-scalable graph computations of the asynchronous Tangle in addition to big data computation technologies, the proposed solution suffices the aforementioned properties.

Minimal end-user client requirements are a side effect of using cluster nodes as service providers, allowing end-users to participate and issue transactions without having to store and handle huge amounts of data but instead providing hashing power.

II. Overview

The protocol maintains a public ledger of transactions that are contained within blocks. In contrast to blockchain-based solutions, blocks reference and thereby approve two previous blocks and indirectly their predecessors to form a directed acyclic graph. A more detailed explanation of the base architecture and other important existing concepts can be found in [4].

As per protocol, a set of blocks is valid if all contained blocks are valid per se and no conflicts exist between any of the contained blocks. When issuing new blocks, it is to be made sure that the set of blocks approved[1] by the new block is valid. Contributing blocks therefore helps securing and validating BigTangle by increasing the confirmation level of all approved blocks.

Participants are connected in a standard peer-to-peer network via a gossip protocol as fall-back solution. In addition to that, BigTangle will employ Apache Kafka data streaming to increase scalability and propagation speed. Network participants are split into two different archetypes: clients and network nodes. Generic end-users and small-time miners can participate as clients that issue transactions with the aid of a network node, using their own hashing power to create blocks and solve the low-difficulty Proof-of-Work themselves, while network nodes maintain a copy of the graph and provide validated tip pairs to build upon.

The network nodes can derive account balances and transaction states from the graph and provide the information to clients in exchange for e.g. hashing power. Furthermore, a network node operator might choose to also participate in the mining process himself by using his available computational power to solve Proof-of-Works for new blocks in addition to validating the blocks.

The server-side validation as described in [4] can be effectively computed by utilizing approximations. The network nodes create their own local view of the Tangle by maintaining helper constructs such as locally confirmed block sets called milestones, based on which

---

[1] Any indirectly or directly referenced blocks (ancestors) of a block are defined as approved by it.

validation is performed. The consequence is that additional conflicts are checked over the naive Tangle conflict checking, resulting in a stricter validity evaluation than without this approximation. Additional information can be found in chapter 3.

Popular solutions to scalable cryptocurrencies include highly complex sharding, voting-based and capital-based Proof-of-Stake or workarounds [7] with the potential for exorbitant fees. Instead, we propose a fully permissionless solution that returns to the well-known Proof of Work approach of the original Bitcoin solution.

Mining rewards serve as a network maintenance incentive instead of fees. Since it is impossible without sharding to allow every device to participate as network nodes while achieving infinite scalability, it is instead intended that network nodes are deployed in sufficiently big computer clusters utilizing state-of-the-art big data technology. Instead of end-users hosting full network nodes and requiring significant computational resources, their clients cooperate with network nodes by e.g. providing mining revenue and building blocks.

Nevertheless, the network stays permissionless and avoids centralized constructs. As long as nodes fulfill the minimum requirements for keeping up with the transaction volume, any node can effectively participate in network validation and mining.

Independent of the mining process, it is also intended for full nodes to exist regardless of mining rewards. For example, super market chains can deploy their own full nodes to process large transaction volumes themselves.

By utilizing Proof-of-Work, we can avoid drawbacks of Proof-of-Stake-based models such as political apathy, regulations and lower financial stability due to missing hardware investments.

To summarize, general properties provided by BigTangle are high hashing power and time-normalized inflation due to mining, infinite scalability, sufficiently fast transaction confirmation times, full decentralization, trustlessness and permissionlessness, feeless transactions and in-principle quantum security. To make use of these properties, BigTangle will natively support custom token issuances, smart contracts and decentralized token exchanges, in turn enabling economically important use cases.

III. Technical Details

In the following, we briefly discuss technical key details. For recapitulation on established concepts, please refer to existing literature such as [4].

i. Implementation

To achieve high scalability, the node implementation is built upon industry standard big data technologies, including but not limited to Apache Hadoop [28], Apache Kafka [29] and Apache Spark [27]. As mentioned before, Kafka is used to achieve efficient propagation, while Spark and its GraphX API are used to achieve efficient and scalable graph processing in the form of Pregel algorithms [30].

ii. Cryptographic Components

BigTangle relies on elliptic curve cryptography for multi-use signatures. The curve used is Bitcoin's Secp256k1. Similarly, the public keys are also Base-58 encoded, allowing Bitcoin users to reuse their addresses in BigTangle. The currently proposed Proof-of-Work hashing function is the Equihash algorithm [5]. The base architecture is quantum resistant as shown in [4].

iii. Transactions and Accounts

The accounting is based on Bitcoin's Unspent Transaction Output (UTXO) model. Users can issue valid transactions as long as they can provide a valid input script for the used UTXOs. The UTXOs use Bitcoins Turing-incomplete stack language allowing for the same set of functionality as found in Bitcoin.

iv. DAG Architecture

As found in [4], we provide a short reasoning on why the Tangle base architecture as mentioned before is qualitatively stable and leads to short confirmation times even for extremely high block volumes.

The following assumptions are made: There exists a valid Poisson point process model for the incoming blocks with constant rate λ, an average block issuance time h, a stationary number of tips L_0, an idealized network latency of h such that any blocks issued at time t become visible as new tips at time $t+h$ and a tip selection in form of a uniform probability distribution over all current tips.

Since approximately λh invisible tips exist due to latency, an equal amount of tips must no longer be tips anymore due to stationarity. This means that the probability of choosing new tips is $r/(r+\lambda h)$ with the current amount of visible tips r. This leads to the mean of $2r/(r+\lambda h)$ tips chosen by a new block. Again, due to stationarity the mean must be equal to 1 such that the new tips replace the old tips without changing the average number of tips. This leads to $r=\lambda h$ or $L_0=2\lambda h$. Assuming L_0 to be large henceforth, this leads to an expected time until first approval of $L_0/(2\lambda)=h$.

Going further, there exists an average time until almost all new blocks approve a block that can be calculated as follows. At that point of time, the block can be considered locally confirmed due to the fact that most new blocks will approve the block.

Let $K(t)$ be the expected amount of approving tips at time t. Since the probability of a tip not being a tip after time h is the tip substitution rate $L_0/(2\lambda h)=1/2$, at time t one half of $K(t-h)$ tips remain unapproved, while the other half is approved at least once. Let $\mathcal{A}$ be the set of tips from time $t-h$ that remain unapproved at time t and $\mathcal{B}$ the set of tips from time $t-h$ that were already approved at time t. Analogously to previous results, the probability p_1 that a new block approves at least 1 block from $\mathcal{B}$ and none from $\mathcal{A}$ such that K increases equals to

$$p_1=\frac{K(t-h)^2}{2L_0}+\frac{K(t-h)}{L_0}(1-\frac{K(t-h)}{2L_0}) \quad (1)$$

and the probability p_2 that a new block approves $\mathcal{B}$ twice such that K decreases is

$$p_2=\frac{K(t-h)^2}{2L_0} \quad (2)$$

The differential equation for $K(t)$ follows: [11]

$$\begin{aligned}\dot{K}(t)&=\lambda\cdot(p_1-p_2)\\&=\lambda\cdot\frac{K(t-h)}{L_0}(2-\frac{K(t-h)}{L_0})\end{aligned} \quad (3)$$

For $K(t)$ up to some $\epsilon L_0 \ll L_0$ the quadratic term can be dropped and with $\lambda h/L_0=1/2$ equation (3) is simplified to

$$\dot{K}(t)=\frac{K(t-h)}{2h} \quad (4)$$

With $K(0)=1$ and $K(t)=e^{\frac{c}{h}t}$ it follows that

$$K(t)=e^{W(\frac{1}{2})\frac{t}{h}}\approx e^{0.352\frac{t}{h}} \quad (5)$$

with $W(\cdot)$ denoting the Lambert W function. The time t_0 until ϵL_0 is reached then evaluates to

$$t_0\approx\frac{h}{W(\frac{1}{2})}\ln\frac{L_0}{\epsilon}\leq 2.84\cdot h\ln L_0 \quad (6)$$

The time between $K(t)$ reaching ϵL_0 and approximately L_0 is neglected, resulting in an average time until confirmation of a block is given by (6). Remembering $L_0=\lambda h$, this result shows that the confirmation time of this architecture scales logarithmically with the block rate such that even very high block rates have a negligible effect on confirmation time.

Although only shown under simplifying assumptions such as uniform tip selection probability and idealized network latency, it can be argued that the above qualitatively holds for MCMC approval strategies and real networks. Simulations have shown that the confirmation time for MCMC approval strategies follows the results shown above [13].

v. Blocks

A block consists of its header and transactions. In addition to fields existing in Bitcoin, the header contains an additional reference to a previous block, a miner address, a type field for the types as seen below and additional data depending on their type. We list all base block types and their use:

Transfer Blocks contain transactions intended to transfer value from one owner to another.

Cross Domain Transfer Blocks contain transactions intended to transfer value from one domain to another. For more information on their intended use, see chapter BigTangle Intranet.

Mining Reward Blocks contain mining reward transactions that are computed in a deterministic fashion. They can require a more difficult Proof-of-Work than standard blocks. More on this in the mining process detailed later.

Token Issuance Blocks are used to issue custom tokens. Token issuances are identified by address plus sequence number and they are legitimized by the corresponding private key signatures. They can be configured to allow or disallow multiple further issuances.

Storage Blocks could be used to store minuscule amounts of user data. User data is identified by address and usage is legitimized by the corresponding private key signatures and the user data can be encrypted. User data is treated as a value object and can be transferred and traded. The BigTangle Mainnet either limits the size of the storage or prohibits such storage functions completely. We thereby create an application layer storage network based on pay for use.

Smart Contract Blocks are used to create a distributed virtual machine for decentralized autonomous corporations, smart contracts and any other distributed applications similar to other alternative blockchains [22][23].

The distributed applications can be implemented in most modern languages and are not limited to using specialized languages. The relevant code and state data is saved in the block as VM containers using technology such as Docker Composer [19] or Kubernetes Containers [16].

The execution changes the state data and creates new blocks in a sequence. As an example, the Mining Reward Process in BigTangle could be implemented as a smart contract and all nodes would execute the same computation for validated mining rewards based on the current data in BigTangle. As another example, a market exchange application could be implemented in such a form that it uses only a specific node for execution without further required validation (but with transparency).

Governance Blocks are used in the governance process. BigTangle participants can cast votes on matters by issuing governance blocks signed with their private keys.

vi. Participants

Participants can take on different roles in the network depending on their available resources and intentions. In the following, possible types of participation are ordered in descending requirements of bandwidth, space and computational power:

Full Nodes keep a copy of the full BigTangle. They can fully participate in the network and provide any requested blocks.

Pruning Nodes maintain a pruned version of BigTangle. Only the most recent blocks in terms of confirmation are kept. The node can fully participate in the mining process.

Clients do not keep a copy of BigTangle. They rely on the nodes to provide them with necessary information to create transactions and blocks. Clients can solve Proof-of-Work for their issued blocks and thereby provide incentives for network nodes to assist them.

vii. Protocol Details

Node Maintenance

In the following, we briefly describe assorted technical details for preparing the Tangle base architecture to achieve scalable operation.

As mentioned before, the BigTangle node implementation locally maintains the milestone, a set of blocks it considers as locally confirmed and thereby in principle finalized. In addition, it maintains additional auxiliary information and block statistics such as rating, cumulative weight, depth, height etc. mostly as indicated in [4]. The milestone update process in simplified form consists of the following steps and is performed as often as possible:

1. Update relevant[2] block statistics.
2. Remove no longer locally confirmed blocks and their dependents[3] from milestone. (Should not happen often.)
3. Find new locally confirmed blocks, use conflict resolution procedure and add a non-conflicting set of those blocks to milestone.

We define as locally confirmed any block that has reached the upper confirmation threshold of $t_{upper} = 70\%$ in terms of rating and is sufficiently deep. We also add a hysteresis to removing blocks to prevent unnecessary reorganizations due to the probabilistic nature of MCMC.

Of particular note is the conflict resolution procedure. It will only find application here if malicious nodes successfully approve conflicting block combinations such that conflicting blocks are considered locally confirmed. In short, we process conflicts in descending order of maximum rating occurring in the conflicts, eliminating all losing candidates by removing them and all their dependents from the milestone or candidate set respectively.

Lastly, we may prune no longer relevant blocks and their statistics to prevent BigTangle from growing indefinitely in terms of storage space.

Validation and Approval Selection

When generating a new block, two previous blocks are required to approve such that no conflict exists in the union of referenced blocks (such that they are valid). To find such conflict-free block pairs, we apply an iterative MCMC algorithm similar to the approach shown in [12] to find single tips. By moving a pair of walkers forward while disallowing any conflicts and giving priority to the walker moving to higher rating, we resolve conflicts similar to the conflict resolution procedure detailed before and reach block pairs consistent with the milestone.

It is important to note that since we use the milestone as a shortcut to evaluating validity of new combinations of unconfirmed blocks, the validation overhead stays approximately

---

[2] Blocks are relevant if they are used in the MCMC random walks of either rating computation or tip selection.

[3] Dependents of a block are all blocks that either approve the block or use an output, token issuances etc. from a transaction of the block.

constant over time under the assumption of constant transaction influx. This coupled with a suitable pruning strategy allows us to avoid increasingly long back-tracing to the genesis block and enables scalability.

A simple example is shown in figure 1. Consider the validation of the black block. Instead of taking the set of all directly or indirectly approved blocks and validating this set, we only take the difference to the milestone (in this example all cyan and black blocks) and then validate this small set against the current milestone. This results in a scalable validity computation.

Note that the milestone block marked with an 'x' is not in the set of directly or indirectly approved blocks but is now also being validated against, meaning that this validation scheme approximates the naive Tangle validity definition by adding additional validation constraints: Instead of only checking for conflicts among all approved blocks, we now check for conflicts among the union of all approved blocks and milestone blocks. These additional constraints do not affect the validation in a negative way, since any milestone blocks are considered locally confirmed and blocks conflicting with the milestone should originally almost never be approved by any new blocks anyways.

In the case of other blocks conflicting with current milestone blocks and achieving higher rating than the milestone blocks, the old milestone blocks are eventually unconfirmed and the new blocks enter the milestone instead.

Mining Process

To incentivise node operation and network maintenance, we introduce a mining process quite similar to the Bitcoin mining process. Since the BigTangle is an asynchronous network and every node sees a different version of the BigTangle, we cannot simply reward what is seen locally since we require an approximately fixed inflation rate. Instead, we introduce a fix point such that every node can calculate the same rewards in a deterministic manner.

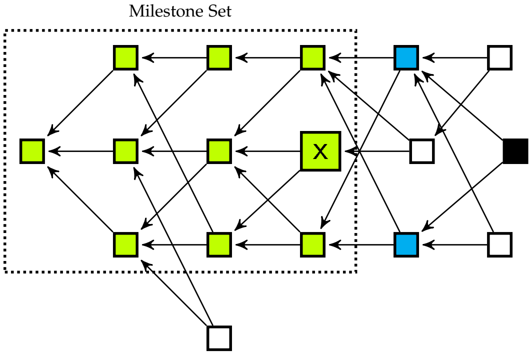

Figure 1: *Example for Validation (black block: block to validate, lime blocks: milestone blocks, white blocks: irrelevant unconfirmed blocks, cyan blocks: to validate vs. milestone, x: additionally confirmed over naive scheme)*

Blocks are divided into height intervals and mining reward blocks are eventually issued after passing the reward height intervals to reward blocks created in the interval. All blocks referenced by the mining reward blocks in the respective interval are considered for compensation and the mining reward block must therefore be in conflict with other such blocks of the same reward height interval.

Using only the blocks approved by the mining reward block, we can compute consistent rewards in a deterministic fashion since we know the referenced subgraph to be unbroken in order for the mining reward block to be considered for confirmation. The calculation of rewards is then done locally. As an example, refer to figure 2 where the red dotted box contains potential reward candidates.

A key problem is deciding on a method to approve mining reward blocks that are consistent and fair according to the nodes

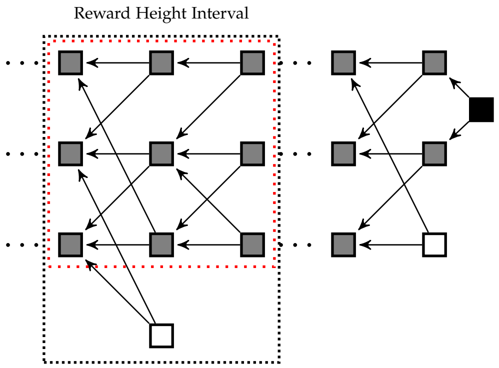

Figure 2: *Example for Mining Process (black block: considered reward block, gray blocks: used in reward calculation, white blocks: not used in calculation, red outline: reward candidates)*

local BigTangle state. The solution is to use the following constraints for approving such mining reward blocks:

All of the approved blocks in the specified height interval must be in the milestone and most of the milestone blocks in the specified height interval must be referenced at the time of reward block reception. To prevent deadlocks due to local inconsistencies, we allow this constraint to be overridden by e.g. sufficient time spent in the milestone, as in that case the majority of the network has accepted the block as valid.

Additionally, we must include countermeasures against non-validating miners' attacks, since otherwise it would be wise to simply build empty transactions upon empty transactions and relink them later.

We punish the blocks with the lowest cumulative weight of their specific height level as seen from the reward block's point view, since attackers will pursue a suboptimal tip selection algorithm in terms of cumulative weight if they avoid validating new blocks. In summary, we calculate cumulative weights for the candidates based on the mining reward block's point of view and then drop out those candidates of lowest cumulative weight in their height.

Finally, the next per-transaction reward is calculated analogously to Bitcoin's difficulty adjustment: By enforcing a monotonous increase of block timestamps 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.

Token Issuance Process

The token issuance process enables a variety of important use cases for the internet of value and supports features as proposed in [24]. We legitimize a new token issuance block by signing their transactions with the token's corresponding multiple private keys. A token is identified by its ID in form of a public key and sequence number.

The issuances can be configured to e.g. disallow further issuances and enforce other custom token rules. Any issuances following another issuance must adhere to the previous issuance's defined rules, e.g. multi-signature checks: the created tokens must be signed by the given number of keys to spent the tokens. The tokens can then be used analogous to BigTangle's system currency.

IV. ATTACK MITIGATION

In the following, we examine a few new attack vectors due to the milestone process. For generic Tangle attack vectors such as double spends, please refer to [4]. We assume that at least two-thirds of the blocks are made by honest miners and validate correctly. For up to one-third of malicious hashing power, although hashing power does not directly correlate with hijacked rating tips, less than one-third of the rating is hijacked by the attackers for most of the relevant blocks since only one-third of the cumulative weight can be hijacked. Further mitigation can be provided by using low-pass filters during rating calculation.

- If percentage p of rating tips maliciously approve double spends, we must ensure that no reorganization occurs, meaning that the lower confirmation threshold must be below $(1 - p)$.
- If percentage p of rating tips maliciously approve a conflict, we must ensure that no network split occurs, meaning that the lower confirmation threshold must be above $(\frac{1-p}{2} + p)$ to prevent the network from having conflicting blocks between their milestones due to a network split where up to 50% of honest miners plus malicious miners would not come to the same consensus as the other honest miners.

The maximum percentage for which such a lower confirmation threshold exists is $p_{max} = 1/3$. The corresponding lower confirmation threshold that follows from the equations is therefore $t_{lower} = 1 - p_{max} = 2/3$.

Parallelizing Proof-of-Work to accumulate highest cumulative weight and in turn gain more rewards 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 building invalid blocks, while building your own subgraph by approving your own blocks only will lead to high orphaning risk due to introducing more than the optimal amount of transitions on your approved blocks.

Trying to relink a pre-built subgraph of higher height to circumvent gaining less cumulative weight is mitigated since rewarded blocks must all have been in the milestone for a while to be accepted by the majority of hashing power. Since there is network latency, it is argued that such a pre-built construct would not allow one to reward oneself only.

V. Mainnet Governance

To assure that the interests of BigTangle participants are safeguarded after its initial release, BigTangle will implement a governance model to achieve clear consensus on its future development.

Currently, it is planned to use a scheme as follows: the stake of stakeholders and hashing power of miners are counted in separate votes and simple majorities on both votes are required to activate BigTangle software updates.

VI. BigTangle Intranet

The BigTangle software can be deployed in private or other trusted environments, allowing one to run private, owned BigTangle networks with different rule sets.

These BigTangle networks are arranged in a hierarchy, i.e. they possess a parent Tangle such as the Mainnet between which a transfer of values is facilitated. For this purpose, each new Tangle has its own interface accounts (addresses) possessed by the private intranet operator from which it is possible to transfer funds into the parent Tangle and vice versa.

A user interested in transferring funds from the parent Tangle into one of its registered child Tangles can transfer tokens to one of the child Tangle's interface accounts, at which point they are either accepted into the child Tangle or returned by the trusted intranet owner.

Inside of such intranets, consensus protocol, transparency, permissiveness and other rules are set by the trusted intranet owner. Transfers of value can be performed internally as it is pleased. For example, in a work agency intranet it would be possible for clients to pay values to work forces in private and in arbitration of the owning work agency.

In general, enterprises and governments can deploy the software internally and e.g. do KYC (Know Your Customer) as well as privacy protection while remaining compatible with BigTangle's Mainnet.

This allows BigTangle to offer a holistic and flexible approach to value management, enabling privacy, transparency and accountability wherever needed by banks, stock exchanges or enterprises.

VII. USE CASES

Projected practical use cases allow the token to derive value and mainly include the substitution of various currently costly and trust-based technical processes. In the following, some important use cases as part of BigTangle's holistic interpretation of the internet of value are briefly explored.

i. Payment

A simple and important use case is payment processing. By providing 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 general populace.

Note that the network hashing power is approximately proportional to the BigTangle internal token market cap and is therefore decoupled from actual transaction volumes, theoretically resulting in downwards unbounded energy upkeep at the cost of increased confirmation times for constant economic risk. Adequate confirmation times can be achieved on a global scale.

ii. Fiat Money

The token issuance protocol can be used to issue bank-backed tokens denoting conventional fiat money. Since the issuance and usage requires no participation in the network, BigTangle is a low cost solution for all parties. Fiat money transactions 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 microdollar range or alternatively via seconds of hashing power due to the departure from winner-takes-it-all, allowing for new business models, e.g. online newspapers with alternatives to commercial advertisement.

v. Supply Chain

Assuming suppliers issuing authenticity tokens, it is trivial to track product authenticity via token transfers. This use case extends into classic supply chain management, allowing the trustless tracking of inventories in supply chains.

VIII. FURTHER INVESTIGATIONS

Proof-of-Work algorithm
ASICs exist for the currently proposed Equihash algorithm. Other Proof-of-Work algorithms may be considered to allow for a more ideal distributed Proof-of-Work on end-user clients.

References

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

[2] BigTangle.net, User Guide: `https://bigtangle.net`

[3] Markov chain Monte Carlo (MCMC), `https://en.wikipedia.org/wiki/Markov_chain_Monte_Carlo`

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

[5] Biryukov, A., & Khovratovich, D. (2017). Equihash: Asymmetric proof-of-work based on the generalized birthday problem. Ledger, 2, 1-30.

[6] nxtforum.org, (2014). DAG: a generalized blockchain. `https://nxtforum.org/proof-of-stake-algorithm/dag-a-generalized-blockchain/`

[7] Poon, J., & Dryja, T. (2016). The bitcoin lightning network: Scalable off-chain instant payments.

[8] Sergio Demian Lerner (2015) DagCoin: a cryptocurrency without blocks. `https://bitslog.wordpress.com/2015/09/11/dagcoin/`

[9] Yonatan Sompolinsky, Aviv Zohar (2013) Accelerating BitcoinâĂŹs Transaction Processing. Fast Money Grows on Trees, Not Chains. `https://eprint.iacr.org/2013/881.pdf`

[10] Yoad Lewenberg, Yonatan Sompolinsky, Aviv Zohar (2015) Inclusive Block Chain Protocols. `http://www.cs.huji.ac.il/~avivz/pubs/15/inclusivebtc.pdf`

[11] Sheldon M. Ross (2012) Introduction to Probability Models. 10th ed.

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

[13] Kusmierz, B., Staupe, P., & Gal, A. (2018). Extracting Tangle Properties in Continuous Time via Large-Scale Simulations. working paper.

[14] Christian Decker, Jochen Seidel, and Roger Wattenhofer. Bitcoin meets strong consistency. In Proceedings of the 17th International Conference on Distributed Computing and Networking, page 13. ACM, 2016.

[15] Christian Decker and Roger Wattenhofer. Information propagation in the bitcoin network. In 13th IEEE International Conference on Peer-to-Peer Computing (P2P), Trento, Italy, September 2013.

[16] Kubernetes Container Orchestration. `https://kubernetes.io/`

[17] Andrew Miller, Yu Xia, Kyle Croman, Elaine Shi, and Dawn Song. The honey badger of bft protocols. In Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, pages 31-42. ACM, 2016.

[18] Yonatan Sompolinsky and Aviv Zohar. Secure high-rate transaction processing in bitcoin. In International Conference on Financial Cryptography and Data Security, pages 507-527. Springer, 2015.

[19] Docker: `https://www.docker.com`

[20] Vitalik Buterin. 2014. Slasher: A Punitive Proof-of-Stake Algorithm. (January 2014).

[21] Protocol Labs, 2018. Filecoin: A Decentralized Storage Network. `https://filecoin.io/filecoin.pdf`

[22] block.one, 2018. EOS.IO Technical White Paper v2. `https://github.com/EOSIO/Documentation/blob/master/TechnicalWhitePaper.md`

[23] ethereum, 2017. A Next-Generation Smart Contract and Decentralized Application Platform `https://github.com/ethereum/wiki/wiki/White-Paper`

[24] Colored coins whitepaper, https://docs.google.com/a/buterin.com/document/d/1AnkP_cVZTCMLIzw4DvsW6M8Q2JC0lIzrTLuoWu2z1BE/edit

[25] Secure property titles with owner authority: https://nakamotoinstitute.org/secure-property-titles/

[26] Mastercoin whitepaper: https://github.com/mastercoin-MSC/spec

[27] Apache Spark. Unified analytics engine for large-scale data processing. https://spark.apache.org/

[28] Apache Hadoop. Reliable, scalable, distributed computing. http://hadoop.apache.org/

[29] Apache Kafka, distributed streaming platform. https://kafka.apache.org

[30] Malewicz, Grzegorz, et al. "Pregel: a system for large-scale graph processing." Proceedings of the 2010 ACM SIGMOD International Conference on Management of data. ACM, 2010.