Crystal Consensus: Dreams to Reality

[Redacted]

February 14, 2024

Abstract

Blockchains hold great potential for connecting society, but struggle to overcome the challenges of their ever-growing use. We propose an enhanced design to address the present scalability hurdles. Leveraging the fairness of fruit-chains, Crystal applies a high-performance block-type coined 'stems' to harness real-time blockchain consensus. We conclude with qualitative analysis showcasing optimal responsiveness and resilience for the crystal blockchain protocol.

1 Introduction

With more than fifteen years having passed since its genesis, Bitcoin's success demonstrates the timeless potential of blockchain technology[10]. Ethereum, a Turing-complete blockchain[2], also features a breadth of applications beyond just crypto-currency. However, as blockchains continue to grow in popularity, their constraints have become increasingly evident.

The potential of blockchains to improve both the efficiency and security of information systems draws interest from public and private sectors alike. Yet, inherent draw-backs of decentralized consensus restricts their overall network capacity, and thus mainstream utility. We present a unified consensus solution. Crystal's protocol is designed to address distributed network scalability while uniquely remaining cohesive with relevant blockchain security frameworks.

1.1 A Next-Generation Blockchain

To preface our problem, we first overview Bitcoin-NG, a "scalable" blockchain[4]. NG separates network functions into distinct block-types. Key-blocks carrying proof-of-work are used solely for leader election. Leaders are thereby tasked to process transactions via micro-blocks, which efficiently omit proof-of-work. Yet, performance still relies on the vertical scalability of the underlying network.

Although streamlining operation, NG remains restricted by hardware constraints of individual nodes. Total network throughput is thereby limited similar to a store with just one check-out lane. Super-nodes (or super-cashiers) could theoretically enhance speed, but this tactic naively centralizes responsibility. Ideally, an element of synergy would allow transactions to occur concurrently.

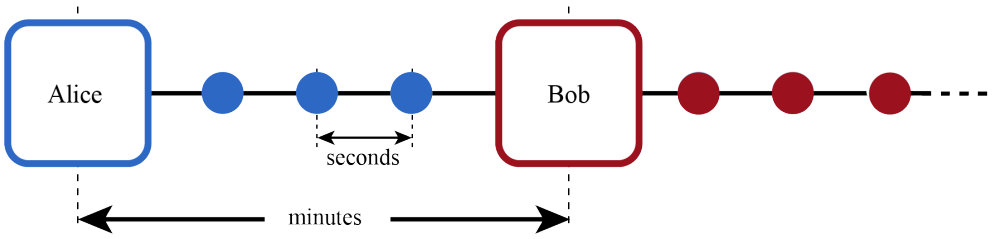


Figure 1: **Bitcoin-NG's micro-chain architecture**. Nodes Alice and Bob self-identify with Blue and Red key-blocks; their suffixed micro-chains depicted with corresponding circles.

We therefore propose an advanced blockchain consensus protocol which allows multiple parallel processing threads (check-out lanes) to be efficiently created and unified.

1.2 A New Leaf

Furthering NG's micro-chain architecture, Crystal integrates a tertiary block-type colloquially denoted as "stems" to provide a highly efficient information nexus. As an artifact of the mining process, stems also provide a valuable source of unbiased entropy. Nomenclature is intuitively derived from "Fruit-Chains"[11], which are further detailed within the System Model section. Fruit-chains, in turn, build upon ideas from the Bitcoin Backbone Analysis[6], which employs the two-for-one 'piggy-backed' mining-schema to host an efficient channel for block-propagation.

Although the main purpose of the micro-chain is to quickly process transactions, the stem does not perform this function directly. Rather, stems serve to rapidly assimilate new micro-blocks. Stems do not introduce new data, but rather reinforce existing information. As a result, they are inherently non-contentious. Stems' neutral property naturally promotes network cohesion. We further leverage this heterogeneous composition to optimally distribute system in order to render the network suitable for real-world workloads.

By providing an information conduit, the stem serves to rapidly process and relay congruent micro-blocks. Functional cross-chain communication proves key for preserving blockchain security characteristics within widely distributed peer-to-peer networks. We analyze consensus performance using the Trees and Chains framework[8]. The proposed state-of-the-art blockchain design notably attains optimal performance while also retaining Nakamoto's[10] original, well-formalized security model.

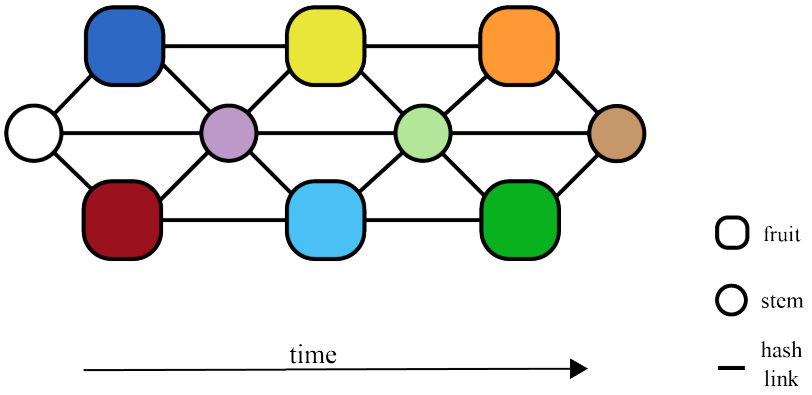


Figure 2: **Heterogeneous micro-block structure.** Alice (Blue) and Bob (Red) create congruent blocks, which are then embed on-chain by a neutral nexus, the (Purple) stem.

Stems (with partial proofs of work) efficiently distribute network load (see Figure 2). New data is processed into micro-blocks (of fruit[11]), which are then multi-cast to the network by staked nodes. As miners work towards finding a successive key-block, they cross-validate recently shared data by conjoining congruent fruit with partial proofs (stems) along the way. As in Bitcoin-NG, new information is only committed to the main-chain (of key-blocks) once a complete proof-of-work has been shared for the entire micro-block structure.

In order to demonstrate the scale-out ability of Crystal's blockchain protocol, we introduce a basic cryptographic sortion. This mechanism is further applied to parallelize transaction processing across a robustly re-configurable[3] validator set. In our analysis, we further examine the fault-tolerance characteristics of our protocol, and evaluate security qualities under adversarial scenarios. We defer quantitative performance testing to future work.

1.3 Road-map

Section 2 features a brief overview of essential preliminaries and related works. Section 3 establishes the system model and assembles the necessary components for composing our dream blockchain. Within section 4 we detail the `Crystal` protocol. Section 5 explores the security implications of a unified weighting both proof-of-work and proof-of-stake blocks in chain-selection, according to the trees and chains framework. In Section 6, we calculate network security characteristics in simulated adversarial scenarios. Finally, Section 7 concludes with an summary of our contribution and potential future optimizations.

2 Building Blocks

Traditionally, blockchains grow in strictly linear fashion, sequentially adding information to a growing tree of knowledge. A linear series of blocks is often referred to as a 'branch'. But in Bitcoin, only the longest branch of blocks is considered in consensus. However, as shown by [15], a single branch can only support so much load before diverging from the block-tree, and detracting from consensus. Blockchain throughput is thereby constrained in a necessary effort to ensure the following security guarantees:

- *Common-Prefix*: Looking back k (security parameter) blocks, nodes share a consistent view of the present block-chain.
- *Chain Quality*: Given a k-length segment of blocks, the majority are from honest nodes.
- *Chain Growth*: New blocks are added to the chain consistently.

2.1 (Under)Mining

Ideal chain-quality (and the potential for performance optimization) is namely undermined by the misalignment of crypto-economic incentives. [5] represents a quintessential example where miners selfishly (yet rationally) deviate from protocol to increase their profit. By "Gaming the system" in this manner, miners can withhold any newly mined blocks in order to get a head start on mining the next block. In addition, through manipulating communication[7], selfish strategies also grant disproportionate influence in chain-selection. As a result, the honest-majority assumption of [10] degrades, requiring a super-majority ($\frac{2}{3}$) of mining-power to ensure an honest-majority ($\frac{1}{2}$) of blocks in the chain[5].

2.2 Haunted Forests

As block-frequency increases in high-performance settings, the likelihood of network bifurcation - where the block-tree diverges into several competing branches - increases as well.[15] However, normally only the longest linear sequence of blocks is accounted for. This results in a scenario where valuable work may be discarded arbitrarily. To accommodate the natural divergence of fast-growing block-trees, Sompolinksy and Zohar propose the GHOST (Greedy Heaviest Observed Sub-Tree) protocol[16], a modification to the longest-chain rule which encourages participants to account for as much chain-growth as possible.

Incorporating valid work from various parts of the block-tree ensures fairer representation of the network. But to do so in confidence, one must have near-perfect knowledge regarding the block-tree. One approach proposes propagating all block-headers, but the absence of linear order renders the network vulnerable to tremendous overhead. Ascertaining the 'main-chain' therefore requires a cumbersome traversal of all recent branches. As a result, both greedy and selfish chain-selection strategies yield sub-optimal performance.

2.3 Hybrid Consensus

An alternative design by Pass and Shi harnesses blockchain security to bootstrap more efficient byzantine fault-tolerant communication methods atop.[12]. The underlying 'main-chain' provides a weakly synchronized global-beacon, which is used to periodically update the network state and maintain record of eligible block-producers. This approach yields nearly ideal performance characteristics:

- Resilience: Optimal responsiveness with $> \frac{3}{4}$ honest active committee members.
- Responsiveness: New transactions are confirmed within actual network propagation delay δ

Hybrid Consensus mechanisms leverage inherent chain-quality to securely elect a committee of recently active nodes. Committees are then responsible for processing the incoming transaction workload for a predetermined duration. Since the network shares a consistent view on present state of the chain, nodes can unanimously agree on committee selection.

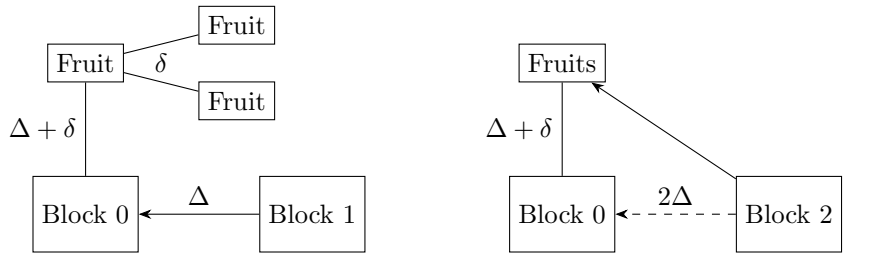
2.4 Sweet Dreams

Sleepy Consensus[13] schedules block-production without proof-of-work. Instead, Sleepy uses a deterministic function to assign block-producers in individual time-steps. This mechanism preserves security assuming a majority of the network is comprised of honest nodes. Block-production is emulated by computing $(\mathcal{P}, t)$ with $\mathcal{P}$ as the party identifier, and t being the current chain-time. This function is invoked each time-step to determine eligibility. The output must satisfy the difficulty threshold $\mathcal{D}$. In essence, $(\mathcal{P}, t) < \mathcal{D}$ signifies eligibility to produce a block, similar to the proof-of-work model introduced by Nakamoto[10].

The absence of proof-of-work does present additional challenges. Whereas honest nodes only make blocks in the present, adversaries can make free use of old time-slots, or attempt front-run the chain with pre-imaged blocks. To account for this additional wiggle-room, Sleepy incorporates more stringent rules for tracking progression of the block-tree.

1. A valid blockchain consists of blocks with strictly increasing time-stamps.
2. Block time-stamps can not jump too far ahead of the present system-time.

Assuming a maximum latency of Δ, any node unresponsive for longer than 2Δ is considered asleep and thereby ineligible for block production. Participant clocks are assumed to 'tick' at the same pace, and be offset by no more than the communication delta.



(a) Key-Block 1 seeds new fruits (b) Key-Block 2 digests propagated fruits

Figure 3: Fruits-chains retroactively finalize transaction-sets

3 Fair Fruits

By providing ideal chain quality characteristics, fruit-chains[11] secure a fairly conducive environment for performance optimization. Like Bitcoin-NG, fruit-chains process data in real-time using a specialized block-type. Transactions are first processed in micro-blocks (called fruit) before being added to the main-chain (of key-blocks). Unlike NG, fruit-chains employ a partial proof-of-work difficulty parameter D_f for their micro-blocks in order to modulate their growth.

While nodes process transaction-sets into fruit, these new micro-blocks are actively shared, resulting in the formation of a fruit-chain along-side the main-chain (see Figure 3a). Throughout the interim between key-blocks, miners reap as many fruit possible into their key-blocks in an effort to maximize their block reward. Similar to the transaction-fee incentive featured in earlier blockchains, this encourages miners to process more transactions in their blocks. Because fruit-chains distinctly encourage participants to promote the fruits of everyone's labor, the mechanism naturally encourages an inclusive consensus environment.

Fruits may be considered as an enhanced micro-block, with some nuance. Whereas typical micro-blocks extend from the chain-tip (or most recent key-block), fruit 'hang' from posterior sections of the block-tree, expanding chain-depth. For reference, fruits also note their current chain-tip and additional metadata regarding neighboring fruit-sets. As fruits propagate, miners compile this new data into cohesive fruit-set before ultimately digesting them into the main-chain (see Figure 3b).

Freshness To allow time for sufficient propagation, fruits are required to 'hang' from the block-tree for a set period before being included in a key-block. The actual duration of the finality delay depends on the worst-case network latency Δ. Fruits are retroactively finalized in the main-chain once they have been acknowledged by a previous key-block. Likewise, a fruit may be discarded from the memory pool if, after a prolonged period, it remains unharvested.

3.1 Waking Snow White

Using Sleepy's[13] consensus mechanism as a foundation, Snow White(SnoW)[3] introduces robust committee reconfiguration functionality for proof-of-stake. Freshly configured validator committees are tasked to process transactions for a ω-length period, referred to as epochs. During each epoch, active members take turns publishing new micro-blocks. For simplicity, we configure the epoch window-length ω to be commensurate with the worst case network delay Δ; this setting frames epochs in the context of key-blocks. SnoW introduces two additional assumptions for secure Proofs-of-Stake.

- Initial Common Knowledge: Implies a known and established record of trust-worthy stake-holders eligible for inclusion in decision-making.
- Epoch-based committee reconfiguration: SnoW assumes a function which samples current chain-state and outputs eligible participants according to present stake-distribution.

SnoW employs a look-back parameter of 2ω to randomly seed the committee initiation process. This reflective exercise ensures committee composition remains up-to-date while also preventing active members from placing strategic seeds within their fruits in an effort to perpetually elect themselves.

4 Crystal Shards

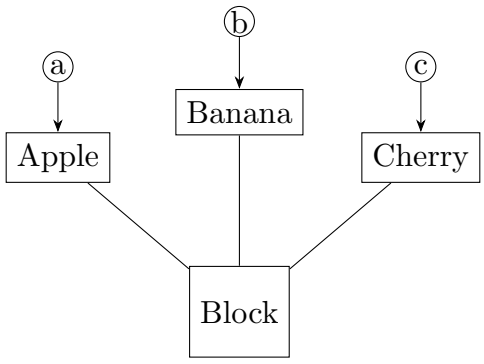


Figure 4: Different seeds, different fruits

To parralelize processing, we further leverage SnoW's robust functionality by spawning several simultaneous chain instances. This is accomplished by splitting the validator pool into various sub-committees, each using a distinct variety of seed to derive unique micro-chains. Shards may then autonomously prioritize their own transaction-sets. The result is a discrete array of partially synchronized fruit-chains at the epoch boundary.

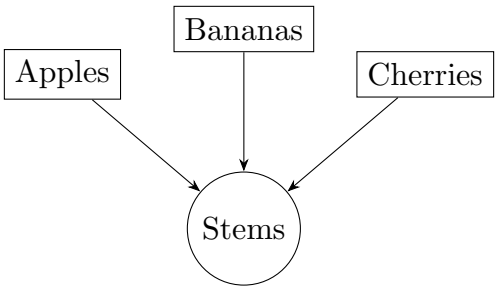


Figure 5: Cross-shard communication flows via stems, enabling various sub-chains to efficiently coexist.

In traditional sharded network design, the potential for adversarial influence is n-fold. For example, adversaries may attempt to strategically 'poison' their fruit seed(s) in an active effort to increase their influence in consensus. Each seed must therefore be sampled by a sufficient number of nodes in order to ensure the shard's overall integrity. The maximum shard-count is therefore limited by the total size of the validator pool.

While our rudimentary approach increases chain capacity, it also broadens the network attack surface proportionate to the extent of sharding present. In addition, when the network is operating at full-capacity, the average node is unlikely to possess sufficient computing power to process every fruit within the epoch window. In this scenario, balancing performance and security becomes non-trivial, as allocating miners more time to complete their cross-reference of fruits introduces additional latency. Furthermore, over-extending SnoW's epoch window deters the robustness of it's functionality by allowing committee composition to become increasingly stale over time.

To overcome the aforementioned technical challenges, we introduce a light-weight on-chain apparatus; this allows us to further distribute responsibility amongst aptly suited parties. In our case, users utilize protocol tokens to create staked-nodes, which efficiently process transactions into fruits. Collectively, miners then proceed to validate this newly shared data by cross-referencing congruent fruits with partial proofs-of-work; this 'stem' establishes a persistent conduit featuring routine convergence opportunities. The presence of stems also asserts any referenced fruit have been determined by miners to be self-consistent.

Embedding fruits with stems provides an extra degree of confidence, since both stake-holders and miners cross-validate each others' work. The prevalence of stems is ultimately a function of the partial difficulty ratio D_f which sets the relative frequency of stems compared to key-blocks. Without loss of generality, we say that each time-step (wherein members produce fruits) is at least δ-length, allowing miners time to both process fruits and produce stems every 2δ. This particular configuration ensures each set of fruits has it's own stem. In practice, stems and fruits coalesce dynamically, leaving the ideal ratio as an area of focus for future optimization.

5 Tree-chains

In this analysis section, we apply the GHOST-compatible Trees and Chains framework[8] to evaluate chain-selection criteria and resulting network security properties. This framework establishes a comprehensive measure of blockchain security, encompassing both tree-based block structures and traditional chains such as Bitcoin and Ethereum. This work not only provides the first formal proof of security for the GHOST protocol, but also greatly simplifies the weighting measure for chain selection in block-trees. Successfully harnessing this approach enables a optimal speed up of transaction processing to O(log)n by permitting concurrent execution of data.

5.1 Unified Weighting Solution

Advanced block-tree architectures require a more holistic approach to chain-selection. To address this need, [14] introduces a universal parameter 'κ' which details the level of chain-depth from which blocks are to be accepted. For instance, excessively high κ settings result in too many outdated transactions being considered, thereby diminishing throughput. Conversely, chains with overly-conservative κ settings such as Bitcoin and Ethereum (wherein κ=0), heavily restrict bandwidth by resorting to the longest-chain rule.

Dreams to Reality The dynamic nature of the Trees & Chains framework provides formal security guarantees for applied block-trees. In our case, we prioritize the tree-chain which bears the most fruit, as it exhibits the broadest support of the network. For reference, fruits begin to propagate after κ=1 or more. Widening the window of opportunity defined by κ enables a greater variety of block-contributions by permitting additional latency for finality. In practice, longer windows grant more time for larger transaction payloads to be processed. Conversely, minimizing the acceptance window restricts network bandwidth through shorter finality requirements for blocks. We adopt the well formalized scenario wherein blocks are accepted from any awake node, as defined by Sleepy[13]. Elected nodes thereby feature a 2Δ window to post new fruit after a seeding period of 1Δ, resulting in a minimum viable κ-parameter of depth-3.

One particularly notable characteristic which emerges from the hybrid mode is the collective influence parties autonomously exert in chain-selection. While traditional mechanisms feature just one group in decision-making, our hybridized blockchain consensus protocol naturally provides incentive for more diverse network representation. This unique trait manifests itself in a powerful system of 'checks and balances', as both stakeholders and miners exert equal influence in consensus. The result is a robustly decentralized mechanism by which new information can be processed, verified and broadcast in efficient manner.

6 Simulations

To analyze Crystal's security characteristics under varying degrees of adversarial influence, we port the 'AttackerSuccessProbability' logic from Bitcoin to capture interplay between Proof-of-Work and Proof-of-Stake blocks. The following code calculates the probability of an attacker successfully overwriting a confirmed transaction depending on the number of blocks (z) added since its inclusion. Python is used instead of C for its increased readability.

```
import math

        class UnifiedBlockchainSimulator:
        def calculate_probability(self, z, q_pow, q_pos):
        # Combine the probabilities for PoW and PoS
        combined_q = (q_pow + q_pos) / 2
        p = 1.0 - combined_q
        lambda_combined = z * (combined_q / p)

        sum = 1.0
        for k in range(z + 1):
        # Unified Poisson distribution for the number of
            blocks added by the attacker
        poisson = math.exp(-lambda_combined)
        for i in range(1, k + 1):
        poisson *= lambda_combined / i

        sum -= poisson * (1 - pow(combined_q / p, z - k))

        return max(0.0, min(sum, 1.0))
```

6.1 Results

We calculate successful attack probability for an adversary which collectively controls a collective a total of 10% of resources within the consensus system.

```
        unified_simulator = UnifiedBlockchainSimulator()

        # Attacker controls 10% of PoW and 10% of active
            protocol stake
        q_pow = 0.1
        q_pos = 0.1

        unified_probabilities = {}
        for z in range(11):  # z from 0 to 10
```

z value	Probability
0	1.00000
1	0.20459
2	0.05098
3	0.01317
4	0.00346
5	0.00091
6	0.00024
7	6.47e-05
8	1.73e-05
9	4.63e-06
10	1.24e-06

Table 1: Probabilities of successful attack for different z values

7 Conclusion

We propose a sophisticated yet intuitive design which holds the potential to unite a vast, globally distributed network in consensus decision-making. First, we provided a conceptual overview of a next-generation blockchain, and present a counter-point against strictly vertical scaling, arguing that decentralization suffers. Necessarily, we introduce the prospect of horizontal scalability. In order to effectuate this design, we followed a multi-faceted approach, learning from strengths and weaknesses of various cohesive works. Inspired by Bitcoin-NG and Fruit-chains, we proposed a novel block-type to unite the distributed networking functionality of Crystal's blockchain. The system notably harnesses emergent synergy between stakeholders and miners to sustain peak performance. Stakeholders assemble transactions into 'fruits', miners then collectively assimilate these batched transaction-sets into the blockchain. In contrast to more recent ad-hoc scalability approaches, we adopt formalized frameworks to reason about security characteristics. Finally, we posit that the described protocol achieves the optimal lower bound for responsiveness in partially synchronous distributed systems, allowing instant transaction confirmation assuming greater than two-thirds honest committee members. The proposal makes ample use of directed acyclic graphs[9] to facilitate transaction processing. Notable improvement in consensus performance is attained by establishing robust incentives for users to maintain functional nodes[1], an approach which was initially suggested during 2015 block-size debates[17].

References

1. Babaioff, M., Dobzinski, S., Oren, S., Zohar, A.: On bitcoin and red balloons. In: Proceedings of the 13th ACM conference on electronic commerce, pp. 56–73 (2012)
2. Buterin, V.: Ethereum: A Next-Generation smart contract and decentralized application platform, (2014). https://ethereum.org/.
3. Daian, P., Pass, R., Shi, E.: Snow white: Provably secure proofs of stake. Tech. rep., Cryptology ePrint Archive, Report 2016/919, 2016. http://eprint. iacr. org/2016/919 (2016)
4. Eyal, I., Gencer, A.E., Sirer, E.G., Renesse, R.V.: Bitcoin-NG: A Scalable Blockchain Protocol. In: 13th USENIX Symposium on Networked Systems Design and Implementation (NSDI 16). USENIX Association, Santa Clara, CA (2016). https://www.usenix.org/conference/nsdi16/technical-sessions/presentation/eyal
5. Eyal, I., Sirer, E.G.: Majority is not enough: Bitcoin mining is vulnerable. Communications of the ACM **61**(7), 95–102 (2018)
6. Garay, J., Kiayias, A., Leonardos, N.: The bitcoin backbone protocol: Analysis and applications. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques, pp. 281–310 (2015)
7. Heilman, E., Kendler, A., Zohar, A., Goldberg, S.: Eclipse Attacks on Bitcoin's Peer-to-Peer Network. In: 24th USENIX Security Symposium (USENIX Security 15), pp. 129–144. USENIX Association, Washington, D.C. (2015). https://www.usenix.org/conference/usenixsecurity15/technical-sessions/presentation/heilman
8. Kiayias, A., Panagiotakos, G.: On Trees, Chains and Fast Transactions in the Blockchain. IACR Cryptology ePrint Archive **2016**, 545 (2016)
9. Lewenberg, Y., Sompolinsky, Y., Zohar, A.: Inclusive block chain protocols. In: Financial Cryptography and Data Security: 19th International Conference, FC 2015, San Juan, Puerto Rico, January 26-30, 2015, Revised Selected Papers 19, pp. 528–547 (2015)
10. Nakamoto, S.: Bitcoin: A peer-to-peer electronic cash system, (2008).
11. Pass, R., Shi, E.: Fruitchains: A fair blockchain. In: Proceedings of the ACM Symposium on Principles of Distributed Computing, pp. 315–324 (2017)
12. Pass, R., Shi, E.: Hybrid Consensus: Efficient consensus in the permissionless model. http://eprint.iacr.org/2016/917.pdf (2016)
13. Pass, R., Shi, E.: The sleepy model of consensus. In: International Conference on the Theory and Application of Cryptology and Information Security, pp. 380–409 (2017)
14. Sompolinsky, Y., Wyborski, S., Zohar, A.: PHANTOM GHOSTDAG: a scalable generalization of Nakamoto consensus: September 2, 2021. In: Proceedings of the 3rd ACM Conference on Advances in Financial Technologies, pp. 57–70 (2021)
15. Sompolinsky, Y., Zohar, A.: Accelerating Bitcoin's Transaction Processing. Fast Money Grows on Trees, Not Chains. In: IACR Cryptology ePrint Archive (2013)
16. Sompolinsky, Y., Zohar, A.: Secure high-rate transaction processing in bitcoin. In: International Conference on Financial Cryptography and Data Security (2015)
17. Unknown, *[bitcoin-dev] Bitcoin XT Fork*, (2015). http://lists.linuxfoundation.org/pipermail/bitcoin-dev/2015-August/010238.html.