

Fast-to-Converge PoW-like Consensus Protocol

Shuyang Tang¹, Sherman S.M. Chow², and Zhiqiang Liu^{1*}

¹ Department of Computer Science and Engineering,
Shanghai Jiao Tong University, Shanghai, China

² Department of Information Engineering,
The Chinese University of Hong Kong, Hong Kong

Abstract. As the fundamental component of various bitcoin-derived blockchains, proof-of-work (PoW) scheme has been widely leveraged to provide consensus for maintaining a distributed public ledger. However, the long confirmation time, and hence the slow convergence rate, is far from satisfactory. Alternative paradigms with improved performance have been proposed. Nevertheless, the modification of the underlying mechanism of proof-of-work is relatively less considered. We find that the slow rate of convergence in original proof-of-work is caused by the insufficient measurement of computational power which only gets one bit of information by judging whether attained hash value is smaller than a given target or not. Due to this, we propose the *Demo-of-Work* (DoW) that assigns the computation work with a score depending on the hash value. We treat an existing blockchain – bitcoin as a global “clock”, according to whose block generations our protocol execution is divided into rounds. Thereby, a synchronization is attained to make sure that each participant takes part in DoW for roughly the same time interval so that the fairness can be formalized and proved. Afterwards, we construct an alternative blockchain – AB-chain with DoW and the “clock”. This blockchain provides a convergence rate significantly faster than the existing PoW-powered blockchains, without compromising communication complexity or fairness.

Keywords: Blockchain, Consensus, Cryptocurrency, Proof-of-Work

1 Introduction

Since 2008, the blockchain mechanism has been providing a consensus protocol for maintaining a distributed ledger in a decentralized manner. The blockchain structure is a chain of blocks, each of which contains transactions to be put into the ledger. To generate a block which includes new transactions, participants perform a *Proof-of-Work* (PoW) [9,15]. Namely, they try to find an admissible *nonce* by brute-force. A nonce is an admissible solution when the hash of the nonce concatenating the hash of the previous block header, (the root of a Merkel tree of) transactions, and other auxiliary data is smaller than a predetermined value. This value is directly related to the *difficulty* of finding a solution. When such a nonce is found, a block is assembled and appended to the end of the blockchain, i.e., it extends the ledger. The process of finding a nonce is referred to as *mining*, due to the award of proposing a block. Thereby, participants are referred to as *miners*.

A fork emerges when a few blocks are mined after the same block, either due to malicious purposes or by coincidence. “Competition” is needed to converge the branches into one and resolve the fork. Honest miners always mine on the longest valid branch, hence malicious forking attempting to tamper the ledger history requires significant work. When any given block is followed by a sufficient number of new blocks, it is considered as *confirmed* since it will never be outraced except for a small probability³. We call it *secure convergence* when the competition ensures such a guarantee.

As more practical applications in need of fast confirmations are built on blockchains, the long confirmation time of blockchains comes to be a serious constraint of future implementations. Sadly, the existing blockchain mechanism reaches secure convergence slowly, i.e., a long confirmation time. From our perspective, the issue of a long confirmation time can be solved from three approaches.

* Corresponding author

³ Different from the cryptographic literature, this paper uses *small probability* to refer to a sufficiently small constant.

The first approach is proof-of-stake that delivers the decision power from miners to stake holders like *Algorand* [14], *Snow White* [6], *Ouroboros* [16], *Ouroboros Praos* [8], and *iChing* [11]. However, PoS schemes are mostly hard to be implemented in practice due to their utter reconstructions of the consensus and that newest results have shown the necessity of proof-of-works [24] in the presence of late spawning.

The second is the optimization leveraged at an application level that considers an existing blockchain as the underlying protocol, like the *Lightning Network* [26] for rapid micropayments, novel scalable blockchain fabrics [28,23,1] that combine a permissioned Byzantine fault-tolerance protocol with a blockchain.

The third approach that we adopt is modifying the very principle of the blockchain itself. Compared with the second approach, instead of a substitution, the third approach is an addition to all improvements from the application level. Due to this, any slight improvements from the underlying blockchain contributes even when protocols of the application level may speed up the convergence by hundreds of times. Unfortunately, to our knowledge, although there are alternative blockchains [10,22] theoretically proposed to reach a greater throughput or a better fairness, improving the convergence rate by an alternative blockchain principle is never considered. We aim at an alternative blockchain rather than a better improvement in the application level. The following roadmap leads to our construction of a new blockchain to provide fast convergence.

- **Demo-of-Work.** We intuitively consider (which is later justified) that the proof-of-work of bitcoin blockchain leads to a slow convergence due to its evaluating work via only one bit of information of whether attaining a hash value smaller than a target. Therefore, we newly propose *Demo-of-Work* (DoW), where a potential function \mathcal{L} (a term borrowed from the Physics literature to provide an intuition of evaluating the underlying works via hashes) assigns a score (the *weight*) to each block according to the block hash, without a hard target of the hash puzzle.
- **Bitcoin As A Global Clock.** Inspired by using blockchain as a publicly verifiable source of randomness [4] and the penalty mechanism from blockchain with applications in secure multi-party computations [2,5,18,19], we for the first time treat an existing bitcoin blockchain as a global clock. Namely, our protocol execution proceeds by rounds and each generation of a bitcoin block ends a round and starts the next round. With such a technique, a synchronization is attained to assure that each participant performs DoW for roughly the same time interval in attempt of assembling each block. Based on this, the fairness is formalized and proved (see Sec. 4).
- **AB-Chain.** Based on two tools above, we build an alternative blockchain (AB-chain) whose ledger extends as bitcoin blockchain grows. Specifically, similarly to bitcoin, our blockchain consists of a linear order of blocks, but each block has a *block weight* assigned by its hash according to a potential function. Honest miners always build blocks onto the chain branch with the greatest *total weight* – a summation of all block weights on the branch, when ambiguity is encountered. However, apart from the previous block, each block should also refer to the bitcoin block on the end of the longest bitcoin chain. In case of a bitcoin fork with multiple newest blocks, any one of them can be chosen. A round starts from the generation of the newest bitcoin block and ends with the generation of the next bitcoin block. During a round, each miner tries to compose a block and find a proper nonce to lessen the block hash as far as possible. It broadcasts the block if the hash is competitive (with a positive block weight no less than that of existing blocks in its view of this round). All peer-to-peer network nodes take part only in the forwarding of competitive blocks.

The *convergence rate* indicates the speed of reaching a secure convergence on a new block. It varies with different potential function \mathcal{L} . To analyze the convergence rate, we propose a novel convergence model and evaluate it experimentally under some selected potential functions. The fairness is guaranteed as long as the potential function is *valid* (monotonously non-increasing regarding the input hash). The communication complexity under certain instantiations is $O(N)$. We prove that the general bound of the communication cost is at most $O(N \log N)$, where N stands for the total number of network nodes. Finally, we pick a potential function to form the final AB-chain, with an $O(N)$ communication cost as bitcoin. This is the first approach to improve the convergence rate by replacing the underlying principle in proof-of-work protocol.

1.1 Related Works

After the proposal of the bitcoin blockchain [21] in 2008, efforts have been devoted to analyzing it. *Bitcoin backbone* [12] has shown two basic properties of the bitcoin protocol – common prefix and chain quality. A model has been built to formally analyze the security and performance of various cryptocurrencies with the existing blockchain [13].

The literature has made various improvements of the throughput of decentralized PoW-based consensus schemes, like *Bitcoin-NG* [10], the side chain [25], and different sharding protocols (e.g., [20]). Moreover, a novel blockchain protocol with better fairness has been proposed by Pass and Shi [22]. More tools have been leveraged to improve the cryptocurrency consensus like the collective signing [17], a permissioned Byzantine fault-tolerance protocol [23,1], or a directed acyclic graph (DAG) [7,27,3].

Organization. We first describe our notations and assumptions as well as our protocol outline in the next section. In Sec. 3, we build a model to discover the rate of secure convergence. In Sec. 4, the communication cost is analyzed and the fairness is shown to be reached under our convergence model, then we choose one potential function elaborately and present the final AB-chain protocol.

2 System Model

2.1 Notations and Assumptions

The notations used in this paper is shown in Tab. 1. To simplify our analysis, we regard the range of the hash function $H(\cdot)$ as $[M]$ where the notation $[M] := \{1, 2, \dots, M\}$ is defined to be all positive integers no greater than M . We denote the global hash rate by T , which is the total number of hash attempts taken by all participants in one round. We assume that T is significantly smaller than M and its value is accessible to all participants. For easier exposition, multiplying a ratio (α, β, δ) with T results in integers.

We denote a block by B . $B.\text{nonce}$ is the nonce solution provided by the proposer and $B.\text{hash}$ is the hash of B . $B.\text{preBlock}^1$ (or just $B.\text{preBlock}$) is the previous block that B follows (or B itself in case that it is the genesis block, i.e., the first block of the chain). $B.\text{preBlock}^k$ ($k \in \mathbb{N}^+ \setminus \{1\}$) is defined recursively,

$$B.\text{preBlock}^k := \begin{cases} B, & \text{isGenesisBlock}(B) \\ (B.\text{preBlock}^{k-1}).\text{preBlock}^1, & \text{otherwise} \end{cases},$$

where $\text{isGenesisBlock}(\cdot)$ is a predicate that returns whether one block is the genesis block. The notation $B.\text{preBlocks} := \cup_{i=1}^{\infty} \{B.\text{preBlock}^i\}$ refers the set of all “ancestor” blocks of B (include all $B.\text{preBlock}^k$ ’s). For an AB-chain block, $B.\text{btcBlock}$ is the block of bitcoin it refers. More notations are shown in Tab. 1.

We assume a peer-to-peer network where each node forwards only competitive nodes in its view, i.e., any block forwarding is aborted if $\mathcal{L}(B.\text{hash}) \leq 0$. When receiving a block B following an AB-chain block B_{-1} that is already followed by B' , each node forwards B only if $\mathcal{L}(B.\text{hash}) \geq \mathcal{L}(B'.\text{hash}) > 0$. We consider that the network is robust, hence any newly generated block on the bitcoin blockchain is revealed to all nodes of AB-chain almost simultaneously.

Honest participants can mine on different branches due to an ambiguity or network delay. We regard that all participants mining on a chain branch that is finally overrun by another branch during a fork.

2.2 Outline of Our Protocol

Chain Structures. The AB-chain structure preserves that of bitcoin, except for few differences (see Fig. 1). Firstly, each block contains a weight corresponding to the block hash. Specifically, the weight of block B equals $\mathcal{L}(B.\text{hash})$, where $\mathcal{L}(\cdot)$ is the potential function that maps a hash value to a specific block

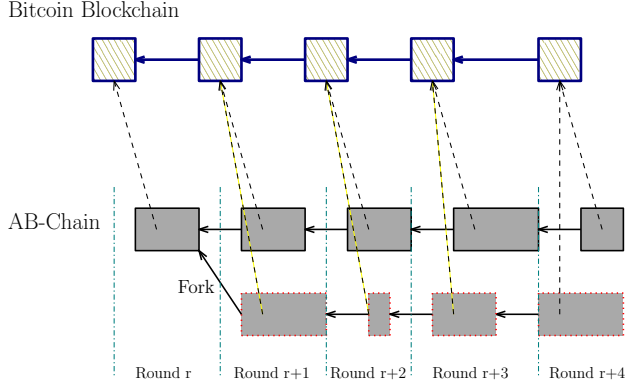


Fig. 1. Outline of The AB-Chain (the length of a block in the graph of our chain indicates the block weight)

weight. Note that, no block with non-positive weight is tolerated. Secondly, each AB-chain block B refers to both the previous block $B.preBlock$ and a bitcoin block $B.btcBlock$ that

$$(B.preBlock).btcBlock = (B.btcBlock).preBlock.$$

Mining. At the beginning of each round, each miner packs hashes of the previous block of AB-chain and the newest block of bitcoin, its collected transactions, along with other auxiliary information into the block record. Afterwards, it tries to find a proper nonce value $B.nonce$ to (as far as possible) minimize the block hash. At the same time, the miner takes part in the forwarding of blocks from others and record the block with the greatest weight w that refers to the same previous block as its. It broadcasts its own block if it finds a nonce that leads to a competitive block weight no less than w . Note that not necessarily every round has a block since it is possible that no block of positive weight is proposed in few rounds.

Fork Resolutions. When multiple blocks are built following the same block, the fork happens. Each honest node builds blocks following the valid branch with the largest total weight. When a fork happens, each branch has a *total weight*, that is the summation of all weights of blocks on this branch. The branch with a total weight greater than others by a *trust gap* Γ is considered to be *confirmed*. That is, we have the confidence that this branch will never be overrun in total weight except for a small probability⁴. In this way, the fork is resolved and the chain convergence is reached. More formal descriptions of the fork resolution are shown in the convergence model of Sec. 3.

We aim to improve the *convergence rate* (the speed of reaching a secure convergence on each block). Specifically, we aim to lessen rounds expected to reach a confidence that one block remains on the chain forever except for a small probability q . We also want $O(N)$ *communication cost*, i.e., the total amount of all communications during the protocol execution. The cost is hard to be measured by any specific quantity, so we measure it with a complexity. Finally, our protocol should be fair. *Fairness* in this paper is defined as the property that the most competitive block of each round with the greatest weight is generated by each party with a probability according to its proportion of hash power.

3 Rate of Secure Convergence

We build a model to describe the convergence rate of all AB-chains with different potential functions. To reach the exact convergence rate of an AB-chain, a Monte Carlo experiment should be conducted under this model. To justify our proposed convergence model, we implement this model and the experiment to evaluate the convergence rate of the bitcoin⁵. As a result, the outcome well matches that of the original bitcoin paper of Nakamoto. After that, we evaluate the convergence rate under different potential functions and present these results.

⁴ Similarly in bitcoin, a trust gap of six block generations assures the safety except for a minor probability of 10^{-3} against the total malicious hash power of 10%.

⁵ Obviously, the existing PoW is a special case of DoW, whose potential function assigns one to all hashes smaller than a predetermined parameter, and zero to others.

Table 1. Table of Notations

Notation	Description
N	the total number of nodes participating in the network
e, γ	the natural constant $e \approx 2.718$ and Euler's constant $\gamma \approx 0.577$
T	the global hash rate, the number of hash attempts taken by all participants in one round
M	the cardinality of the range of $H(\cdot)$, for example, $M = 2^{256}$ for SHA-256
α	the fraction of the adversary hash power within the global rate
$\mathcal{L}(\cdot)$	a potential function that returns the block weight of a hash value input
\mathcal{H}_{\min}^T	an oracle that returns the minimal hash value for T hash attempts
$\text{Rev}(y)$	the inverse function of $y = x \ln \frac{M}{x}$, well-defined for $0 < x < M/e$
$\text{sgn}(x)$	a function that returns 1 for positive x , -1 for negative x , and 0 for 0
$\Gamma_{\mathcal{L},q}^{\alpha,T}$	the (α, T, q) -trust gap, see Def. 6
$\Lambda_{\mathcal{L},q}^{\alpha,T}$	the inverse of the (α, T, q) -convergence rate, see Def. 7
D	a quantity $D := M/T$

3.1 Convergence Model

To facilitate the description of our convergence model, we propose the notion of “minimal hash value oracle”, which provides an equivalent simulation of the mining process of each participant. Specifically, this oracle inputs the number of hash attempts T_0 , and returns which is the least value among T_0 uniform random selections from the range of the cryptographic hash function $H(\cdot)$. This simulates the mining since the event of “mining an AB-block” is essentially “having the least hash value of T_0 hash attempts small”. Now we formalize this definition.

Definition 1 (Minimal Hash Value Oracle). For a positive integer T_0 , oracle $\mathcal{H}_{\min}^{T_0}$ outputs the minimal hash value in T_0 hash attempts. For a fixed T_0 , $\forall i \in [M]$, it is equivalent to a discrete probabilistic distribution with

$$\Pr \left[h \leftarrow \mathcal{H}_{\min}^{T_0} \mid h \leq i \right] = 1 - \left(1 - \frac{i}{M} \right)^{T_0}.$$

Then, we formally propose our defined potential function $\mathcal{L}(\cdot)$, to assign a weight to each block regarding the block hash h , to measure the work required to reach a hash value no greater than h . In addition, we ask for each potential function $\mathcal{L}(\cdot)$ to be *valid*, i.e., $\mathcal{L}(h)$ should be monotonously non-increasing by h , since smaller the least hash value is, more hash attempts are done in expectation.

Definition 2 (Valid Potential Function). In the AB-chain consensus, a potential function $\mathcal{L} : [M] \rightarrow \mathbb{R}^+$ is called *valid* if and only if for all integer $1 \leq i < M$, $\mathcal{L}(i) \geq \mathcal{L}(i+1)$.

Afterwards, we propose terminologies regarding the convergence (i.e., resolving of chain forks) one-by-one. To begin, we define the chain weight and offer the formal chain competition criteria.

Definition 3 (The Total Weight of A Chain). For a chain of blocks $\text{chain} = (B_0, B_1, \dots, B_\ell)$ (from the genesis block to the newest valid block), with $B_{i-1} = B_i.\text{preBlock}^1$ for all $i \in [\ell]$, its total weight is $\text{weight}(\text{chain}) := \sum_{i=1}^{\ell} \mathcal{L}(B_i.\text{hash})$.

To measure the convergence rate regarding the establishment of a potential function $\mathcal{L}(\cdot)$, we first introduce the notion of the “trust gap”. A sufficient trust gap Γ is required to make sure that as long as one chain branch has a total weight greater than all others’ with such a gap, we have the confidence that this branch reaches a secure convergence, i.e., will never be outraced except for a small probability q , and the fork is resolved. The quicker chain competitions are done (i.e., fewer rounds expected to attain such a trust gap), the better a convergence rate is reached. Fig. 2 provides an intuition of fork resolutions.

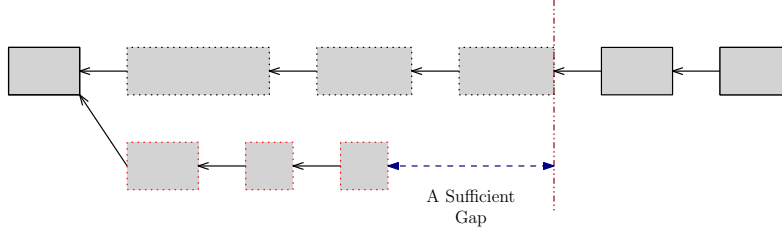


Fig. 2. AB-Chain with Fork Resolutions

Definition 4 (The Chain Competition). When two chains of blocks exist in case of a chain fork, for instance, $\text{chain}_1 = (B_0, B_1, \dots, B_\ell)$ and $\text{chain}_2 = (B'_0, B'_1, \dots, B'_{\ell'})$, assuming the fork starts from the k^{th} block (i.e., $B_i = B'_i$ for all natural number $i < k$), chain_1 outraces chain_2 if and only if

$$\sum_{i=k}^{\ell} \mathcal{L}(B_i.\text{hash}) - \sum_{j=k}^{\ell'} \mathcal{L}(B'_j.\text{hash}) > \Gamma,$$

which is equivalent to

$$\text{weight}(\text{chain}_1) - \text{weight}(\text{chain}_2) > \Gamma,$$

for a certain gap Γ . (The determination of this gap will be provided in Def. 6.)

Definition 5 ((R, Δ, α, T, q)-Confidence). For an AB-chain consensus with potential function $\mathcal{L}(\cdot)$ and round number R , it achieves (R, Δ, α, T, q)-confidence if

$$\Pr \left[\begin{array}{l} x_1, x_2, \dots, x_R \leftarrow \mathcal{H}_{\min}^{\alpha, T} \\ y_1, y_2, \dots, y_R \leftarrow \mathcal{H}_{\min}^{(1-\alpha)T} \end{array} \middle| \sum_{i=1}^R \mathcal{L}(x_i) - \sum_{j=1}^R \mathcal{L}(y_j) > \Delta \right] \leq q.$$

In later parts, α is the proportion of the hash rate of the adversary, T is the global hash rate, and q is a small probability.

As mentioned, a trust gap is set to assure that any branch with a weight greater than others by such a gap has a confidence that other branches can no longer gain a total weight greater than its except for a small probability q . We formally describe how this gap is obtained.

Definition 6 ((α, T, q)-Trust Gap). For an AB-chain with potential function $\mathcal{L}(\cdot)$, we denote (α, T, q)-trust gap $\boxed{\Gamma_{\mathcal{L}, q}^{\alpha, T}}$ as the least Δ satisfying its (R, Δ, α, T, q)-confidence for all $R \in \mathbb{N}$.

Once such a trust gap is achieved by one chain branch over others, the fork is resolved and the secure convergence is reached. Finally, the convergence rate is the inverse of the expected⁶ rounds required to reach a safety gap.

Definition 7 ((α, T, q)-Convergence Rate). For an AB-chain consensus with potential function $\mathcal{L}(\cdot)$, its (α, T, q)-convergence rate is $\boxed{1/\Lambda_{\mathcal{L}, q}^{\alpha, T}}$, where

$$\Lambda_{\mathcal{L}, q}^{\alpha, T} := \frac{\Gamma_{\mathcal{L}, q}^{\alpha, T}}{\mathbb{E} \left[\begin{array}{l} x \leftarrow \mathcal{H}_{\min}^{\alpha, T} \\ y \leftarrow \mathcal{H}_{\min}^{(1-\alpha)T} \end{array} \middle| \mathcal{L}(y) - \mathcal{L}(x) \right]},$$

where the denominator is the expected gap formed in one round.

To facilitate later comparisons, we will only use $\Lambda_{\mathcal{L}, q}^{\alpha, T}$ (which is essentially the inverse of the convergence rate) to signify the convergence rate in later parts. The greater $\Lambda_{\mathcal{L}, q}^{\alpha, T}$ is, a slower convergence rate is attained. $\Lambda_{\mathcal{L}, q}^{\alpha, T}$ is denoted by Λ_α when no ambiguity exists (in which case $1/\Lambda_\alpha$ is the convergence rate).

⁶ Not exactly the mathematical expectation, but it simplifies descriptions.

Demo-of-Work. We now present the Demo-of-Work (DoW) scheme compliant with the above convergence model. In this scheme, we evaluate computation work by using potential function instead of checking whether attained hash value is smaller than a given target or not. Specifically, we assign a score (the weight) to each block according to the block hash and the potential function. This allows exploiting more information related to computational power, which helps measure the work accumulated over a chain (i.e., the overall weight of the chain) more accurately. Further, the accumulation of work in DoW scheme grows faster than that in the original PoW protocol if a nice potential function is adopted, thus leading to a faster convergence rate. The experiments shown below also confirm that the convergence rate in DoW is significantly improved compared with that of traditional PoW. Interestingly, the PoW scheme can be regarded as a special instantiation of the DoW scheme.

3.2 Testimony of Our Convergence Model

With a slight difference⁷ of not referring to an existing blockchain, we view the bitcoin blockchain [21] as an AB-chain with potential function

$$\mathcal{L}(h) = \begin{cases} 1, & h \leq D \\ 0, & \text{otherwise} \end{cases}.$$

The difficulty parameter D here is determined to have one block with the weight of 1 mined in expectation for one round, i.e., $T \cdot (D/M) = 1$. Mining a bitcoin block corresponds to mining an AB-chain block with the weight 1.

To show the reliability of our model, an experiment is conducted to calculate the convergence rate under different assumptions on the adversary hash rate. Details of the experiment have been shown in Appendix A by setting $q = 0.001$. For its results in Tab. 2, the first column reflects the adversary hash rate, the second and the third columns are numbers of rounds needed to confirm a block provided by two independent Monte Carlo experiments of our model. We can observe from the fourth and fifth columns that the experimental result is coherent with that of Nakamoto in 2008 [21].

Table 2. Experimental Results for Convergence Rate on Bitcoin Blockchain

Adversary Hash Rate α	Experiment I	Experiment II	Average Result	Result of Bitcoin [21]
0.10	4.0258	4.0128	4.0193	5
0.15	6.9477	7.0094	6.9785	8
0.20	10.2896	8.1600	9.2248	11
0.25	16.1332	15.1540	15.6436	15
0.30	24.5419	25.2738	24.9079	24
0.35	44.1077	41.7835	42.9456	41
0.40	105.5780	89.7551	97.6666	89
0.45	363.5730	311.9380	337.7555	340

4 AB-Chain: A DoW-driven Blockchain with Fast Convergence Rate

When devising a DoW-powered blockchain with fast convergence rate, communication cost and fairness need to be fully taken into account since they are the vital factors for efficiency and security of the blockchain. In the following, we can also see that the choice of potential function $\mathcal{L}(\cdot)$ is closely related to the communication cost and fairness. Our analysis of these two factors gives the bounds of communication cost for valid potential functions, and proves that fairness always holds as long as the potential function is a monotonously non-increasing function. This provides direction as

⁷ A coherence on results even with “noise” from such a difference further justifies the reliability of our convergence model.

well as confidence in determining a "good" potential function which balances the convergence rate and communication cost without compromising the fairness. Then with this direction, we choose a "good" potential function experimentally and come up with AB-chain accordingly.

4.1 Communication Cost

Since hashes of amount $O(1)$ can be found to satisfy $h < D$ by all participants each round ($\frac{D}{M} \times T = 1$), we can infer that as long as $\mathcal{L}(h) \leq 0$ holds for all $h > kD$ with some constant k , the communication cost should remain $O(kN) = O(N)$ (N is the total number of nodes of the network), since the expected number of proposed blocks will be no greater than k times that of

bitcoin. For instance, let $\mathcal{L}(h) = \begin{cases} 2, & h \leq D \\ 1, & D < h \leq 2D \\ 0, & h > 2D \end{cases}$, $\mathcal{L}(h) \leq 0$ holds for all $h > 2D$, and the overall

communication cost is bounded by $O(2N) = O(N)$. Although a $O(N)$ communication cost is guaranteed for certain cases, such a complexity is not reached by all cases of the generalized model. As an analysis of the general model, we can prove that the overall communication cost will not exceed $O(N \log N)$.

A General Bound. We assume that each miner generates a block with a unique block hash, and their blocks are proposed in turn. Each block is successfully proposed (and cause an $O(N)$ communication burden to the network) only if its block has a hash smaller than all previous proposed blocks, or no node forwards its block.

Theorem 1. *For any valid potential function $\mathcal{L}(\cdot)$, assuming each node $i \in [N]$ has a nonce solution of hash value h_i , and proposes successfully (i.e., causing an $O(N)$ network burden) only if $\mathcal{L}(h_i) > \max_{j=1}^{i-1} \mathcal{L}(h_j)$. Then, the overall communication cost is bounded by the complexity $O(N \log N)$.*

Proof. Without a significant twist on results, we assume all nodes submit solutions with different hash values. We denote indicator \mathcal{I}_i as 1 if $\mathcal{L}(h_i) > \max_{j=1}^{i-1} \mathcal{L}(h_j)$, and 0 otherwise. Similarly, we denote indicator \mathcal{J}_i as 1 if $h_i < \min_{j=1}^{i-1} h_j$, and 0 otherwise. Since $\mathcal{L}(\cdot)$ is a valid potential function, we can infer that $\mathcal{L}(h_i) > \max_{j=1}^{i-1} \mathcal{L}(h_j) \Rightarrow h_i < \min_{j=1}^{i-1} h_j$, and hence $\mathcal{I}_i \leq \mathcal{J}_i$ for each $i \in [N]$.

The overall communication cost is $\sum_{i=1}^N \mathcal{I}_i$ times $O(N)$, which should be smaller than $\sum_{i=1}^N \mathcal{J}_i$ times $O(N)$. We symbolize $\sum_{i=1}^n \mathcal{I}_i$ as F_n for each $n \in [N]$. Next, we have

$$\mathbb{E}[F_n] = \left(\frac{1}{n} \cdot 0 + \frac{1}{n} \mathbb{E}[F_1] + \frac{1}{n} \mathbb{E}[F_2] + \cdots + \frac{1}{n} \mathbb{E}[F_{n-1}]\right) + 1$$

for each $n \in [N]$. Denoting $S_n := \sum_{i=1}^n \mathbb{E}[F_i]$, we have

$$S_n - S_{n-1} = \frac{1}{n} S_{n-1} + 1$$

for each $n \in [N] \setminus \{1\}$, and $S_1 = 1$. Denoting $T_n := \frac{S_n}{n+1}$, we next have

$$T_n = T_{n-1} + \frac{1}{n+1},$$

for each $n \in [N] \setminus \{1\}$, and $T_1 = \frac{S_1}{2} = \frac{1}{2}$. Thereby $T_n = \sum_{i=1}^n \frac{1}{i+1}$, and $S_n = (n+1) \sum_{i=1}^n \frac{1}{i+1}$. Finally,

$$\begin{aligned} \mathbb{E}[F_n] &= S_n - S_{n-1} = (n+1) \sum_{i=1}^n \frac{1}{i+1} - n \sum_{i=1}^{n-1} \frac{1}{i+1} \\ &= (n+1) \cdot \frac{1}{n+1} + \sum_{i=1}^{n-1} \frac{1}{i+1} = \sum_{i=1}^n \frac{1}{i} \approx \ln n - \gamma \end{aligned}$$

for each $n \in [N] \setminus \{1\}$. In conclusion, the overall communication cost is bounded by $O(N \times \ln N) = O(N \log N)$.

4.2 The Fairness

The fairness considers the most competitive block with the greatest weight for a single round is proposed by a participant according to its hash power.

Definition 8 (The Fairness). *For an AB-chain consensus scheme with the potential function $\mathcal{L}(\cdot)$, it achieves the fairness if and only if*

$$\Pr \left[x \leftarrow \mathcal{H}_{\min}^{\beta \cdot T}, y \leftarrow \mathcal{H}_{\min}^{(1-\beta)T} \mid \mathcal{L}(x) \geq \mathcal{L}(y) \right] \geq \beta - \epsilon$$

holds for any party holding β rate of global hash power (where $\epsilon = o(\frac{T}{M})$ is a negligible component), T is the number of hash attempts done by all participants for one round.

We formally prove that as long as the potential function $\mathcal{L}(\cdot)$ is valid, the fairness is achieved.

Theorem 2. *Basic fairness of an AB-chain consensus always holds as long as the potential function $\mathcal{L}(\cdot)$ is a monotonously non-increasing function.*

Proof. Suppose that $\mathcal{L}(\cdot)$ is monotonously non-increasing, then

$$x \leq y \Rightarrow \mathcal{L}(x) \geq \mathcal{L}(y).$$

Hence

$$\begin{aligned} \Pr \left[x \leftarrow \mathcal{H}_{\min}^{\beta \cdot T}, y \leftarrow \mathcal{H}_{\min}^{(1-\beta)T} \mid \mathcal{L}(x) \geq \mathcal{L}(y) \right] \\ \geq \Pr \left[x \leftarrow \mathcal{H}_{\min}^{\beta \cdot T}, y \leftarrow \mathcal{H}_{\min}^{(1-\beta)T} \mid x \leq y \right], \end{aligned}$$

and

$$P_{inf} \leq \Pr \left[x \leftarrow \mathcal{H}_{\min}^{\beta \cdot T}, y \leftarrow \mathcal{H}_{\min}^{(1-\beta)T} \mid x \leq y \right] \leq P_{sup},$$

where

$$\begin{aligned} P_{inf} &= \sum_{i=1}^M \frac{1}{M} \cdot \beta T \cdot \left(\frac{M-i}{M}\right)^{\beta T-1} \cdot \left(\frac{M-i}{M}\right)^{(1-\beta)T} \\ &= \frac{\beta T}{M^T} \sum_{i=1}^M (M-i)^{T-1} = \frac{\beta T}{M^T} \sum_{i=0}^{M-1} i^{T-1} \\ &= \frac{\beta T}{M^T} \int_0^M x^{T-1} dx - O\left(\left(\frac{T}{M}\right)^2\right) \\ &= \frac{\beta T}{M^T} \cdot \frac{M^T}{T} - O\left(\left(\frac{T}{M}\right)^2\right) \\ &= \beta - O\left(\left(\frac{T}{M}\right)^2\right), \end{aligned}$$

and that

$$\begin{aligned} P_{sup} &= \sum_{i=1}^M \frac{1}{M} \cdot \beta T \cdot \left(\frac{M-i+1}{M}\right)^{\beta T-1} \cdot \left(\frac{M-i+1}{M}\right)^{(1-\beta)T} \\ &\lesssim \frac{\beta T}{M^T} \cdot \frac{(M+1)^T}{T} \\ &= \beta \cdot \left(1 + \frac{1}{M}\right)^T \approx \beta \cdot e^{T/M}. \end{aligned}$$

Since we may assume that

$$T \ll M,$$

we have $\Pr \left[x \leftarrow \mathcal{H}_{\min}^{\beta \cdot T}, y \leftarrow \mathcal{H}_{\min}^{(1-\beta)T} \mid x \leq y \right] \gtrsim \beta - O\left(\left(\frac{T}{M}\right)^2\right)$, and so forth

$$\Pr \left[x \leftarrow \mathcal{H}_{\min}^{\beta \cdot T}, y \leftarrow \mathcal{H}_{\min}^{(1-\beta)T} \mid \mathcal{L}(x) \geq \mathcal{L}(y) \right] \geq \beta - O\left(\left(\frac{T}{M}\right)^2\right).$$

This achieves our defined fairness since $O\left((T/M)^2\right) = o(T/M)$ is negligible.

4.3 The Choice of Potential Function and the Resulting AB-Chain

With the above analysis, we have the rough direction in determining a “good” potential function which reaches nice balance between the convergence rate and communication cost while not sacrificing the fairness. Under this direction, we select representatives of monotonously non-increasing functions as potential functions delicately according to the gross type of function curve, and implement independent experiments for different type of potential functions via the Monte Carlo method (simulations) under our convergence model. For each chosen potential function, the estimated communication complexity and the experimental results of the convergence rate under adversary assumptions of 10%, 20%, 30%, and 40% are shown in the respective columns of Tab. 3.

Then, we decide $\mathcal{L}(h) = \begin{cases} 2, & h \leq D \\ 1, & D < h \leq 2D \\ 0, & h > 2D \end{cases}$ due to its faster convergence rate and a $O(N)$ com-

munication complexity. For instance, according to the result of Tab. 3, for the adversary assumption of $\alpha = 10\%$, 4 rounds are sufficient for the safety of $q = 10^{-3}$, while 6 rounds for the bitcoin. Also, for the $\alpha = 30\%$ case, 15 rounds of AB-chain reaches roughly the same security as that of 25 rounds to bitcoin.

5 Conclusion

We proposed demo-of-work that assigns a score to evaluate the computation work on each block more accurately by using potential function. Further, we used an existing reliable blockchain as a global clock that divided the protocol execution into rounds. Based on two tools above, we have constructed a novel blockchain protocol with a faster convergence rate as well as satisfactory communication cost and fairness. This protocol for the first time changed the underlying mechanism of proof-of-work and could be applicable to any blockchains adopting PoW as the blockchain protocol.

This work spawns various future research directions, especially regarding the potential function. A systematic approach of finding an optimal potential function is desired. A formal analysis of incentives from the perspective of game theory is expected. More relationship between the potential function and the convergence rate needs to be discovered. Finally, this model is expected to be put into practice and attested with real and empirical data.

References

1. I. Abraham, D. Malkhi, K. Nayak, L. Ren, and A. Spiegelman. Solidus: An incentive-compatible cryptocurrency based on permissionless byzantine consensus. *CoRR*, abs/1612.02916, 2016.
2. M. Andrychowicz, S. Dziembowski, D. Malinowski, and L. Mazurek. Secure multiparty computations on bitcoin. In *2014 IEEE Symposium on Security and Privacy, SP 2014, Berkeley, CA, USA, May 18-21, 2014*, pages 443–458, 2014.
3. Anton Churyumov. Byteball: A decentralized system for storage and transfer of value, 2016.
4. I. Bentov, A. Gabizon, and D. Zuckerman. Bitcoin beacon. *CoRR*, abs/1605.04559, 2016.
5. I. Bentov and R. Kumaresan. How to use bitcoin to design fair protocols. In *Advances in Cryptology - CRYPTO 2014 - 34th Annual Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2014, Proceedings, Part II*, pages 421–439, 2014.
6. I. Bentov, R. Pass, and E. Shi. Snow white: Provably secure proofs of stake. *IACR Cryptology ePrint Archive*, 2016:919, 2016.
7. X. Boyen, C. Carr, and T. Haines. Blockchain-free cryptocurrencies: A framework for truly decentralised fast transactions. *Cryptology ePrint Archive*, Report 2016/871, 2016.
8. B. David, P. Gazi, A. Kiayias, and A. Russell. Ouroboros praos: An adaptively-secure, semi-synchronous proof-of-stake blockchain. In *Advances in Cryptology - EUROCRYPT 2018 - 37th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tel Aviv, Israel, April 29 - May 3, 2018 Proceedings, Part II*, pages 66–98, 2018.
9. C. Dwork and M. Naor. Pricing via processing or combatting junk mail. In *Advances in Cryptology - CRYPTO '92, 12th Annual International Cryptology Conference, Santa Barbara, California, USA, August 16-20, 1992, Proceedings*, pages 139–147, 1992.

10. I. Eyal, A. E. Gencer, E. G. Sirer, and R. van Renesse. Bitcoin-NG: A scalable blockchain protocol. *13th USENIX Symposium on Networked Systems Design and Implementation, NSDI 2016, Santa Clara, CA, USA, March 16-18, 2016*, pages 45–59, 2016.
11. L. Fan and H. Zhou. iChing: A scalable proof-of-stake blockchain in the open setting (or, how to mimic nakamoto’s design via proof-of-stake). *IACR Cryptology ePrint Archive*, 2017:656, 2017.
12. J. A. Garay, A. Kiayias, and N. Leonardos. The bitcoin backbone protocol: Analysis and applications. In *Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part II*, pages 281–310, 2015.
13. A. Gervais, G. O. Karame, K. Wüst, V. Glykantzis, H. Ritzdorf, and S. Capkun. On the security and performance of proof of work blockchains. *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, October 24-28, 2016*, pages 3–16, 2016.
14. Y. Gilad, R. Hemo, S. Micali, G. Vlachos, and N. Zeldovich. Algorand: Scaling byzantine agreements for cryptocurrencies. In *Proceedings of the 26th Symposium on Operating Systems Principles, Shanghai, China, October 28-31, 2017*, pages 51–68, 2017.
15. M. Jakobsson and A. Juels. Proofs of work and bread pudding protocols. In *Secure Information Networks: Communications and Multimedia Security, IFIP TC6/TC11 Joint Working Conference on Communications and Multimedia Security (CMS ’99), September 20-21, 1999, Leuven, Belgium*, pages 258–272, 1999.
16. A. Kiayias, A. Russell, B. David, and R. Oliynykov. Ouroboros: A provably secure proof-of-stake blockchain protocol. In *Advances in Cryptology - CRYPTO 2017 - 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 20-24, 2017, Proceedings, Part I*, pages 357–388, 2017.
17. E. Kokoris-Kogias, P. Jovanovic, N. Gailly, I. Khoffi, L. Gasser, and B. Ford. Enhancing bitcoin security and performance with strong consistency via collective signing. In *25th USENIX Security Symposium, USENIX Security 16, Austin, TX, USA, August 10-12, 2016.*, pages 279–296, 2016.
18. R. Kumaresan and I. Bentov. Amortizing secure computation with penalties. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, October 24-28, 2016*, pages 418–429, 2016.
19. R. Kumaresan, V. Vaikuntanathan, and P. N. Vasudevan. Improvements to secure computation with penalties. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, October 24-28, 2016*, pages 406–417, 2016.
20. L. Luu, V. Narayanan, C. Zheng, K. Baweja, S. Gilbert, and P. Saxena. A secure sharding protocol for open blockchains. *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, October 24-28, 2016*, pages 17–30, 2016.
21. S. Nakamoto. Bitcoin: A peer-to-peer electronic cash system, 2008.
22. R. Pass and E. Shi. Fruitchains: A fair blockchain. In *Proceedings of the ACM Symposium on Principles of Distributed Computing, PODC 2017, Washington, DC, USA, July 25-27, 2017*, pages 315–324, 2017.
23. R. Pass and E. Shi. Hybrid consensus: Efficient consensus in the permissionless model. In *31st International Symposium on Distributed Computing, DISC 2017, October 16-20, 2017, Vienna, Austria*, pages 39:1–39:16, 2017.
24. R. Pass and E. Shi. Rethinking large-scale consensus. In *30th IEEE Computer Security Foundations Symposium, CSF 2017, Santa Barbara, CA, USA, August 21-25, 2017*, pages 115–129, 2017.
25. J. Poon and V. Buterin. Plasma: Scalable autonomous smart contracts, 2017.
26. J. Poon and T. Dryja. The bitcoin lightning network: Scalable off-chain instant payments, 2016.
27. Y. Sompolinsky, Y. Lewenberg, and A. Zohar. Spectre: A fast and scalable cryptocurrency protocol. *Cryptology ePrint Archive*, Report 2016/1159, 2016.
28. M. Vukolic. The quest for scalable blockchain fabric: Proof-of-work vs. BFT replication. In *Open Problems in Network Security - IFIP WG 11.4 International Workshop, iNetSec 2015, Zurich, Switzerland, October 29, 2015, Revised Selected Papers*, pages 112–125, 2015.

A Detailed Protocols of The Simulation Experiment

For each instantiation of the generalized model with a potential function $\mathcal{L}(\cdot)$, an experiment can be performed to reveal its convergence rate with a Monte Carlo method. In our conducted experiment, $M = 2^{20}$, $T = D = 2^{10}$ and $q = 10^{-3}$ are chosen, the algorithms listed below are executed (starting from the main function, Algo. 7). After the execution, the main function returns the expected number of rounds required to form the safety gap.

1. **Preparation**(α, T) prepares two arrays to provide outcomes of two discrete cumulative distribution functions. Specifically, $\mathbf{aCDF}[i]$ is the probability of having a hash value no greater than i found by the adversary. $\mathbf{hCDF}[i]$ is similarly the probability of having a hash value no greater than i found by honest nodes.
2. **GetH**(**CDF**) returns a random number according to a distribution of a cumulative distribution function recorded in the array **CDF**.
3. **SimAttack**($\Delta, \mathbf{aCDF}, \mathbf{hCDF}$) models the behaviour of the adversary attempt of forming a new chain of blocks with a total weight greater than the honest one by a certain gap Δ .
4. **Test**($\Delta, q, \mathbf{aCDF}, \mathbf{hCDF}$) performs “SimAttack” for sufficiently enough times, to show (via Monte Carlo method) whether the probability of adversary’s successfully performing an attack (and overrunning the honest one by a total weight of Δ) is smaller than q .
5. **FindMinGap**($q, \mathbf{aCDF}, \mathbf{hCDF}$) utilizes a binary search to find the minimal gap δ that can ensure that the adversary can catch up with the honest chain by a total weight of δ only with a negligible probability q .
6. **Expc**(**CDF**) returns the expected block weight attained by either honest parties or the adversary by another Monte Carlo experiment.
7. **Main**(α, T, q) is the main function that returns the (inverse of) convergence rate A_α of the blockchain with potential function $\mathcal{L}(\cdot)$, i.e., the expected number of rounds required to form the safety gap.

In our final experiment, $\text{NUM_TEST_SAMPLE} = 100000$, $\text{NUM_TEST_BLOCK} = 200$, and $\epsilon = 10^{-4}$.

Algorithm 1 Preparation(α, T)

```

1: aCDF[0] := 0
2: hCDF[0] := 0
3: for  $i := 1$  to  $M$  do
4:   aCDF[i] :=  $1 - (1 - \frac{i}{M})^{\alpha T}$ 
5:   hCDF[i] :=  $1 - (1 - \frac{i}{M})^{(1-\alpha)T}$ 
6: end for
7: return (aCDF, hCDF)

```

Algorithm 2 GetH(CDF)

```

1:  $x := \text{random}(0, 1)$ 
2: return binary_search(CDF,  $x$ )

```

Algorithm 3 SimAttack(Δ , aCDF, hCDF)

```

1:  $S_1, S_2 := 0$ 
2: for  $i := 1$  to NUM_TEST_BLOCK do
3:    $S_1 := S_1 + \mathcal{L}(\text{GetH}(\text{aCDF}))$ 
4:    $S_2 := S_2 + \mathcal{L}(\text{GetH}(\text{hCDF}))$ 
5:   if  $S_1 - S_2 > \Delta$  then
6:     return true
7:   end if
8: end for
9: return false

```

Algorithm 4 Test(Δ, q , aCDF, hCDF)

```

1: succ := 0
2: for  $i := 1$  to NUM_TEST_SAMPLE do
3:   if SimAttack( $\Delta$ , aCDF, hCDF) then
4:     succ := succ + 1
5:   end if
6: end for
7: return (succ <  $q * \text{NUM\_TEST\_SAMPLE}$ )

```

Algorithm 5 FindMinGap(q , aCDF, hCDF)

```

1: left := 0
2: right := SUFFICIENT_LARGE
3: while right - left >  $\epsilon$  do
4:    $x := (\text{left} + \text{right})/2$ 
5:   if Test( $\Delta, q$ , aCDF, hCDF) then
6:     right :=  $x$ 
7:   else
8:     left :=  $x$ 
9:   end if
10: end while
11: return right

```

Algorithm 6 Expc(CDF)

```

1:  $S := 0$ 
2: for  $i := 1$  to NUM_TEST_SAMPLE do
3:    $S := S + \mathcal{L}(\text{GetH}(\text{CDF}))$ 
4: end for
5: return  $S/\text{NUM\_TEST\_SAMPLE}$ 

```

Algorithm 7 Main(α, T, q)

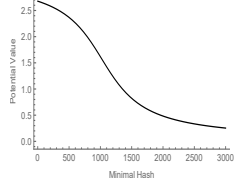
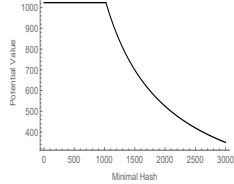
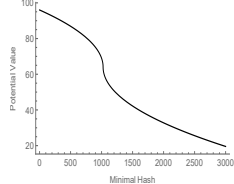
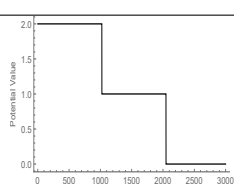
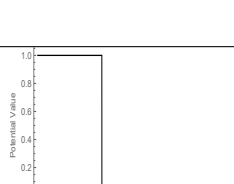
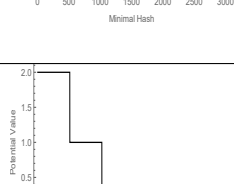
```

1: (aCDF, hCDF) := Preparation( $\alpha, T$ )
2:  $\Delta_0 := (\text{Expc}(\text{hCDF}) - \text{Expc}(\text{aCDF}))$ 
3: convergence := FindMinGap( $q$ , aCDF, hCDF)/ $\Delta_0$ 

4: return convergence

```

Table 3. Experiments on Few Establishments of The Potential Function

Potential Function	Figure	$\Lambda_{0.1}$	$\Lambda_{0.2}$	$\Lambda_{0.3}$	$\Lambda_{0.4}$	Communication Complexity
$\mathcal{L}(h) = \frac{\pi}{2} - \arctan \frac{h-D}{2D}$		1.68	4.13	11.01	48.66	$O(N \log N)$
$\mathcal{L}(h) = \min\{\frac{M}{h}, \frac{M}{D}\}$		2.00	4.67	11.60	49.37	$O(N \log N)$
$\mathcal{L}(h) = 2\sqrt{D} - \text{sgn}(h-D) \cdot \sqrt{ h-D }$		2.25	5.21	13.66	54.81	$O(N \log N)$
$\mathcal{L}(h) = \begin{cases} 2, & h \leq D \\ 1, & D < h \leq 2D \\ 0, & h > 2D \end{cases}$		2.63	6.09	14.83	66.17	$O(N)$
$\mathcal{L}(h) = \begin{cases} 1, & h \leq D \\ 0, & h > D \end{cases}$		4.02	9.22	24.91	97.67	$O(N)$
$\mathcal{L}(h) = \begin{cases} 2, & h \leq \frac{D}{2} \\ 1, & \frac{D}{2} < h \leq D \\ 0, & h > D \end{cases}$		4.89	11.60	26.02	103.18	$O(N)$