Fast Difficulty Adjustment in Proof-of-Work Consensus*

Juan Garay†[1], Aggelos Kiayias[2], and Yu Shen[3]

[1]Texas A&M University, garay@tamu.edu
[2]University of Edinburgh and IOG, aggelos.kiayias@ed.ac.uk
[3]The Hong Kong Polytechnic University, shenyu.tcv@gmail.com

Abstract

One of the main hallmarks of proof-of-work (PoW) consensus protocols is their ability to adjust the difficulty of the PoW, so that it accurately reflects the level of participation and hence maintains security in a setting where protocol participation is unknown and may over time change dramatically. Importantly, this enables the protocol to retain its fundamental characteristics (such as the regularity of dispensing new tokens) irrespectively of the number of parties running the protocol (also known as "miners") at any given time.

The speed with which difficulty can be adjusted is a fundamental feature of a blockchain protocol: The faster the rate with which the difficulty is adjusted, the more agile the protocol is in the face of fluctuating participation. In this work, we put forward, for the first time, a blockchain protocol that performs difficulty adjustment in *constant time*; prior provably secure designs only offered protocols with, at best, poly-logarithmic overhead for difficulty adjustment. Our construction is based on the parallel-chain approach and a new target recalculation function that adjusts mining difficulty making use of information from all chains in the past epoch via a novel application of approximate-agreement techniques that may be of independent interest.

*An abridged version of this paper appears in *Proc. CRYPTO 2026*.

†Research supported in part by NSF Award CCF-205569. Any opinion, findings, and conclusions or recommendations expressed in this material are those of the author and do not necessarily reflect the views of the National Science Foundation.

Contents

1 Introduction **3**
- 1.1 Our Results . . . 3
- 1.2 Related Work . . . 5

2 Technical Overview **6**

3 Preliminaries **9**
- 3.1 Protocol Execution Model . . . 10
- 3.2 Dynamic Participation . . . 10
- 3.3 Consensus and State Machine Replication . . . 11

4 PoW Blockchains with Fast Difficulty Adjustment **11**
- 4.1 Our Parallel-Chain Framework . . . 11
- 4.2 A New Chain Difficulty Rule . . . 13
- 4.3 Difficulty Adjustment in Constant Rounds . . . 16

5 Security Analysis **19**
- 5.1 Additional Notation and Preliminary Results . . . 19
- 5.2 Ideal Convergence Events . . . 23
- 5.3 'Good' Properties Hold . . . 28
- 5.4 Consensus and SMR with Fast Difficulty Adjustment . . . 34

A Mathematical Facts **38**

B Preliminaries (Cont'd) **39**

C Glossary **41**

1 Introduction

Proof-of-Work (PoW)-based blockchain protocols [Nak08] were first studied in a cryptographic setting by Garay *et al.* [GKL15], who showed that all incoming transactions can be "settled" after $\mathcal{O}(\mathsf{polylog}\kappa)$ rounds, where κ is the security parameter, albeit in a static setting where a fixed (yet unknown) set of parties run the protocol with a fixed mining target. Later on, it was shown in [GKL17] that by adopting Bitcoin's target recalculation function, the blockchain protocol is able to adapt to a dynamic environment where the number of parties fluctuates during every window of $\mathcal{O}(\mathsf{polylog}\kappa)$ rounds, while preserving asymptotically the same settlement time. Note that in practice, following Nakamoto's parameterization, epochs (of target recalculation) in Bitcoin are 2016 blocks, or roughly two weeks in real time. Different target recalculation functions were then proposed in [GKS25] and in an unpublished manuscript [CEM+20] by Chan *et al.*, offering asymptotically the same results as in [GKL17] but even allowing the dynamic participation to be driven adaptively by the adversary.

Difficulty adjustment is fundamentally a time-based operation. Irrespective of the exact technical details, the core of the approach is invariably as follows: A time window is determined somehow on chain, within which the total work invested by the network is estimated. This amount is then compared with the expected work assuming no change in participation has taken place since the last recalculation point. Subsequently, the difficulty is adjusted accordingly to the differential of these two quantities—actual vs. expected—subject to upper and lower boundary constraints.

A significant barrier towards achieving anything in a "fast" manner[1] in Nakamoto-style protocols, is the fact that in constant time, concentration bounds are not applicable, and furthermore, anti-concentration arguments suggest that bad events (such as for example the adversary generating more PoWs compared to honest parties) can easily happen with constant probability. As a result, determining the difficulty level, or even settling a transaction, can be perilous since the chain observed by an honest party can be entirely adversarially constructed and hence misleading.

A fundamental workaround for this challenge came about with the "parallel blockchains" paradigm [FGKR18, BKT+19, YNHS20, FGKR20, GKS24, GKS25], where multiple blockchains are mined at once using a suitable composition method originating in the 2×1 PoW method proposed in [GKL15]. The core idea is that, due to parallelism and independence, in constant time, some concentration arguments across multiple blockchains can still apply and hence enable the parties to make progress. (Observe that this will fundamentally require at minimum a poly-logarithmic number of blockchains in κ.) For example, "Prism" [BKT+19] is a PoW-based blockchain protocol that provides fast (i.e., $\mathcal{O}(1)$ rounds in expectation) settlement—albeit for *non-conflicting* transactions—in the static setting, while the "Chain-King Consensus" protocol [GKS24] is a PoW-based blockchain that achieves fast settlement for all transactions. The protocol runs with a fixed number of parties, which was later improved in [GKS25] by incorporating Bitcoin's target recalculation function independently on each parallel chain, thus adapting to the dynamic setting with party fluctuation, but again in $\mathcal{O}(\mathsf{polylog}\kappa)$-round windows.

1.1 Our Results

In this paper we present a new PoW-based blockchain protocol following the parallel-chains approach with a constant target recalculation epoch. The reason prior work in the parallel-chain setting has not managed to make it fast is two-fold. On one hand, allowing each chain to adjust

[1] "Fast" in our exposition refers to achieving objectives in constant time, measured in number of rounds, as opposed to in time in some way growing with security parameter κ.

difficulty independently runs into the same anti-concentration problem we identified earlier: A single chain in constant time may experience an adversarial majority throwing its target recalculation significantly off. On the other hand, if one tries to coordinate a common target across all chains, to exploit the fact that concentration can work across chains (as the logic goes for fast settlement, for example), this creates a cross-chain dependency that potentially ruins the benefit of the parallel chain approach: Recall that parallel chains are attractive exactly because *chain independence* enables the amplification of a desirable event across chains. In our construction we manage to circumvent these obstacles by adopting an *approximate-agreement* approach (cf. [DLP+86]) to target recalculation that can be seen as a compromise between the above two viewpoints. The target that will be admissible by any single chain is not a singular, deterministically calculated value given chain data. Instead, we allow parties to determine difficulty levels approximately, without insisting on same target from different honest parties even within any one single chain. The key benefit of this relaxation is that in a constant number of rounds, for those chains in the parallel chain set that advanced with a reasonable pace and incorporated a good number of honest blocks (a property that in itself, the honest parties are oblivious to), the chain's intended recalculated target, had it been calculated in the standard way, it would fall within the range of targets that are admissible based on our approximate agreement. Interestingly, it is also worth pointing out that our notion of approximate agreement differs from previous formulations in that it is applied to the *ratio* between the maximum and minimum honest outputs, as opposed to the classical application that bounds their absolute difference.

Armed with our parallel blockchain protocol we demonstrate how it is possible to derive PoW-based consensus and state machine replication (SMR). The main benefit of our resulting SMR protocol is in that party fluctuation can be constrained to change in constant intervals for the first time.

Our results and their comparison to prior work is summarized in Table 1.

Protocol	**Settlement**	**Dynamic**	**Epoch Duration**
[GKL15]	$\mathcal{O}(\mathsf{polylog}\kappa)$	✗	-
[GKL17]	$\mathcal{O}(\mathsf{polylog}\kappa)$	✓	$\omega(\log\kappa)$
[GKS22, CEM+20]	$\mathcal{O}(\mathsf{polylog}\kappa)$	✓	$\omega(\log\kappa)$
[BKT+19]	Exp. $\mathcal{O}(1)^*$	✗	-
[GKS24]	Exp. $\mathcal{O}(1)$	✗	-
[GKS25]	Exp. $\mathcal{O}(1)$	✓	$\omega(\log\kappa)$
This work	Exp. $\mathcal{O}(1)$	✓	$\mathcal{O}(1)$

Table 1: *Comparison of PoW-based blockchain protocols in terms of their settlement time and ability to handle dynamic participation.* **: Running time here refers to the optimistic case when no transactions are "double-spent."*

1.2 Related Work

Epoch-based target recalculation functions. Bitcoin adjusts its mining difficulty for every "epoch" consisting of 2016 blocks (roughly equivalent to 2 weeks assuming a 10-minute block production time). Blocks in the same epoch share the same mining target. At the onset of each epoch, the amount of difficulty to be adjusted is set proportionally to the ratio of the time elapsed during the last epoch and the expected epoch duration. Additionally, there is a bound τ ("dampening factor," cf. [GKL17]) that limits the relative amount of difficulty that can be adjusted each time (in Bitcoin, $\tau = 4$). More formally, let T_{ep} denote the target of blocks in epoch ep, f denote the ideal block generation rate and m denote the number of blocks in an epoch (thus, the ideal epoch duration is m/f). The target value of the next epoch is computed as

$$T_{\mathsf{ep}+1} \triangleq T_{\mathsf{ep}} \cdot \min\left\{ \max\left\{ \frac{r_{\mathsf{ep}+1} - r_{\mathsf{ep}}}{m/f}, 1/\tau \right\}, \tau \right\}, \tag{1}$$

where r_{ep} denotes the timestamp of the first block in epoch ep.

In [GKS22], a "reversed" version of Bitcoin's target recalculation function is proposed, which adjusts the difficulty for every epoch of *fixed* duration. This function avoids the explicit use of block timestamps, and adjusts the mining target by counting the number of blocks generated in the previous interval, with the adjustment being inversely proportional to the ratio between the actual number of blocks and their expected number. Formally, let Λ denote the ideal number of blocks in an epoch, and let Λ_{ep} be the actual number of blocks in epoch ep, then

$$T_{\mathsf{ep}+1} \triangleq T_{\mathsf{ep}} \cdot \min\left\{ \max\left\{ \Lambda/\Lambda_{\mathsf{ep}}, 1/\tau \right\}, \tau \right\}. \tag{2}$$

While these epoch-based target recalculation functions are designed for the single-chain setting, they can also be applied independently on each chain in the multi-chain setting (cf. [FGKR20, GKS25]).

On difficulty-raising attacks. It has been shown that a hard-coded upper bound on the relative amount of difficulty that can be adjusted each time is crucial to the security of blockchains [GKL17]. Without such a bound, the so-called "difficulty raising attack" [Bah13] applies which allows an adversary with arbitrarily small mining power to create private forks that are "heavier" than honest ones, by building chains of increasing difficulty and leveraging the high variance. Notably, Garay *et al.* [GKL17] showed that when the protocol is parameterized with an epoch duration of $\Theta(\mathsf{polylog}\kappa)$ rounds and executions of length $\mathsf{poly}(\kappa)$ are considered, where κ is the security parameter, raising difficulty attacks succeed with probability negligible in κ.

Approximate agreement. In the approximate agreement (AA) problem [DLP+86], parties with real-valued inputs are required to output values satisfying (i) Validity where all outputs stay in the convex hull of honest inputs, and (ii) ϵ-Agreement where the absolute difference among honest outputs is bounded by ϵ. AA has been extensively studied in the literature, see e.g. [GLZ25] for an overview. Notably, AA can be solved in *constant* time (as opposed to the linear round-complexity lower bound of consensus protocols [DS83]), thus making it a core building block in clock synchronization protocols, both "classical" (e.g., [LL22]) and blockchain-based [GKS25].

Organization of the paper. The rest of the paper is organized as follows. In the next section, we provide a technical overview of our construction and proof strategy. In Section 3, we introduce the model and define some basic notation. Our new protocol with fast difficulty adjustment is presented in Section 4, while its security analysis is presented in Section 5. Mathematical facts, more detailed specifications and complementary material are presented in Appendices A and B, respectively.

2 Technical Overview

We consider an environment that can adaptively introduce and remove parties (miners) as long as in any window of *constant* number of rounds, there is a bound on the fraction of party fluctuation. We divide the protocol time into consecutive, non-overlapping intervals of *constant* number of rounds, which we call "epochs." Our protocol runs m-ary parallel blockchains, where their mining procedure is bound using $m\times1$ PoWs [GKS24][2].

***Approximate* agreement on target recalculation values.** Since it is impossible to reach full agreement over the mining difficulty in a constant number of rounds, we adopt the natural relaxation that honest parties do *not* agree exactly on one block target; rather, in each epoch they will approximately agree on their mining targets. We achieve this goal by proposing a new target recalculation mechanism that we now explain.

Recall the epoch-based target recalculation function proposed in [GKS22], where the next-epoch target $T_{\mathsf{ep}+1}$ is computed based on the current target T_{ep} adjusted proportionally to the ratio ρ_{ep} between the expected number of blocks and their actual number in the current epoch—i.e., $T_{\mathsf{ep}+1} \triangleq T_{\mathsf{ep}} \cdot \rho_{\mathsf{ep}}$ (cf. Equation 2). Our new target recalculation mechanism consists in letting each party P find its own reference target $T^{\mathsf{P}}_{\mathsf{ep}}$ and reference ratio $\rho^{\mathsf{P}}_{\mathsf{ep}}$, and then compute its own mining target for the next epoch $T^{\mathsf{P}}_{\mathsf{ep}+1} := T^{\mathsf{P}}_{\mathsf{ep}} \cdot \rho^{\mathsf{P}}_{\mathsf{ep}}$. Once P computes $T^{\mathsf{P}}_{\mathsf{ep}+1}$ at the beginning of a new epoch, based on its local view of the parallel chains in the previous epoch, P will proceed to mine blocks under $T^{\mathsf{P}}_{\mathsf{ep}+1}$ throughout the new epoch.

Next, we explain how the reference targets and ratios are selected. Let $(\mathcal{C}_1, \ldots, \mathcal{C}_m)$ be the local parallel blockchains of an honest party. In our protocol, parties will additionally mine a second type of "input-block" (whose mining procedure is m-ary as well, and the i-th input-block mining procedure is bound with the i-th parallel chain using 2×1 PoWs), while trying to generate and include input-blocks into their corresponding chains. In other words, from the last epoch of each chain $\mathcal{C}_i$, an honest party can extract a set of input-blocks B_i. Then, for each set B_i, an honest party can further select a target T_i which is the ***median*** target among all input-blocks, and a ratio ρ_i which is the ratio between the expected total difficulty and the difficulty of input-blocks that has been actually accumulated. Note that, since input-blocks might be mined under different targets, the expected total difficulty is defined as the product of the median target T_i and a pre-defined protocol parameter Λ (the expected number of blocks in the first epoch). These block targets and ratios form two m-ary vectors $(T_1, \ldots, T_m)$ and $(\rho_1, \ldots, \rho_m)$, respectively. Finally, parties apply an approximate agreement function $\mathsf{ApproxAgree}$ over these two vectors, yielding $T^{\mathsf{P}}_{\mathsf{ep}}$ and $\rho^{\mathsf{P}}_{\mathsf{ep}}$, and thus $T^{\mathsf{P}}_{\mathsf{ep}+1}$. For a formal definition, refer to Equations (7) to (9) in Section 4.3. See Figure 1 for an illustration of this procedure.

Let us now elaborate on the $\mathsf{ApproxAgree}$ function (formally defined in Equation (5)). For an m-ary vector $(T_1, \ldots, T_m)$ and an additional protocol parameter $\eta \in (0, m/3)$, the function first orders $(T_1, \ldots, T_m)$ non-decreasingly, and then removes the η largest and η smallest targets from both sides. Then, $\mathsf{ApproxAgree}$ picks a new vector of representative targets, by splitting the remaining $m - 2\eta$ targets into chunks of η elements and then selecting the first element in each chunk. $\mathsf{ApproxAgree}$ returns the average among representative targets, which yields $T^{\mathsf{ep}}_{\mathsf{P}}$. The vector $(\rho_1, \ldots, \rho_m)$ is processed analogously, resulting in $\rho^{\mathsf{ep}}_{\mathsf{P}}$.

Intuitively, the $\mathsf{ApproxAgree}$ function has the following desired property: Even if each honest party invokes $\mathsf{ApproxAgree}$ with different views of local parallel chains, as long as the overlapping part[3] of their views consists of a sufficiently large subset of chains (and thus input-block sets), then

[2] Recall that $m\times1$ PoWs, $m = \mathsf{polylog}\ \kappa$, are an extension of the 2×1 PoW mining technique from [GKL15].

[3] We note that such overlapping subset is achieved in an *oblivious* manner: honest parties would not be able to

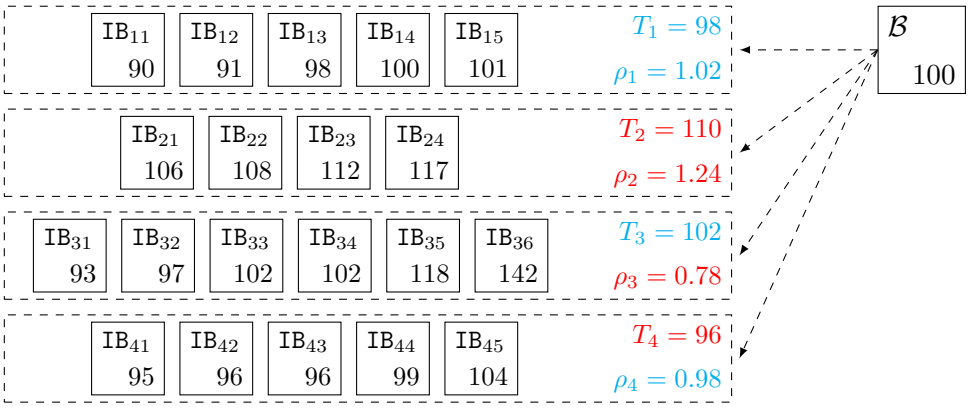

Figure 1: *An illustration of the input-block sets referred by a next-epoch block $\mathcal{B}$. Numbers on the bottom right of blocks refer to their mining targets. We have 4 parallel blockchains and set the expected number of blocks $\Lambda = 5$ and the approximate-agreement parameter $\eta = 1$. Text in red means that median-target/ratio is dropped, while text in cyan shows those values used in computing the reference targets and ratios for $\mathcal{B}$.*

their outputs from ApproxAgree will be close to each other. More precisely, consider the subset of inputs that honest parties obliviously agree on. If the ratio between the maximum and minimum in this subset is bounded by Φ, then, the ratio between the maximum output and the minimum output of any two honest parties will be less than Φ—i.e., ApproxAgree concentrates the inputs. We elaborate on this property of ApproxAgree in Lemma 1. Looking ahead, in our analysis we make an inductive argument that, if at the onset of an epoch, honest parties mine under targets that are close to each other, then in the new epoch they will mine under new targets that are well bounded.

To facilitate our new target recalculation mechanism, we let parties include their local view of parallel blockchains into the block header, by inserting m block hashes of the last blocks on their local parallel chains at the end of the previous epoch. (This will consume κ bits as in our protocol block hashes are of κ/m bits.) Hence, given a new incoming block $\mathcal{B}$, anyone can verify its validity by first checking the existence of these blocks (and their validity), and then apply the ApproxAgree function to verify if the hash of $\mathcal{B}$ is smaller than its prescribed mining target. Accordingly, our parallel blockchain construction accepts blocks with varying targets, conditioned on their passing the above verification and extending an existing block. (In fact, there are more validation rules which we will explain in Section 4.)

Approximate agreement on chain difficulty. Our ultimate goal is to show that our parallel blockchain construction is secure in a dynamic environment where the number of parties fluctuates in every window of constant number of rounds. However, in addition to the impossibility of full agreement over the mining target, another challenge that emerges with epochs of constant duration is that adversarial attacks on a single chain would succeed with constant probability. In particular, in constant time, a raising difficulty attack [Bah13] always succeeds with constant probability.

We note that previous works utilizing parallel blockchains to construct for example a consensus (Byzantine agreement) protocol [GKS24] are not readily applicable in this context, as they are either restricted to the static setting or dynamic but assuming a slow party fluctuation [GKS25]; in the former case the mining target is fixed throughout the execution, while in the latter it is adjusted independently on each chain for every epoch of super-logarithmic rounds.

reach full agreement over the overlapping subset.

Suppose we run the parallel construction from [GKS25] and naïvely shorten the epoch duration to constant rounds and apply our new target recalculation function. We end up with the following problem: There might be an epoch where the adversary manages to produce a large number of blocks on, say, the first chain (recall that he can always do so with constant probability). The adversary can then arrange the block timestamps for the next epoch, thus gaining an advantage (similar to a pre-mining attack). Consequently, target recalculation in the next epoch may not function correctly, as blocks in that epoch are not a correct reflection of the computational power.

We address these technical obstacles by augmenting our protocol with a new chain difficulty rule, which is again an application of approximate agreement. Specifically, we define the difficulty of a chain $\mathcal{C}$ in a recursive manner. Let ep denote the epoch of the last block on a chain $\mathcal{C}$. The difficulty of $\mathcal{C}$ consists of two parts: The first part is the sum of difficulties of all blocks in $\mathcal{C}$ in epoch ep, and the second part results from applying $\mathsf{ApproxAgree}$ over the set of chain difficulties $(d_1, \ldots, d_m)$, where d_i is the difficulty of a chain in the previous epoch specified by the first block $\mathcal{B}$ of $\mathcal{C}$ in ep (recall that from $m\times1$ PoW, $\mathcal{B}$ has to point to m chains in the previous epoch). We formally define this chain difficulty rule in Equation (6) in Section 4.2. Then, in our protocol, parties still use the heaviest chain selection rule to filter chains to work on, where 'heaviest' is defined according to our new chain difficulty rule. Honest parties keep extending m-ary parallel chains, with each chain being the heaviest in their local view.

Intuitively, the advantages of our new chain difficulty rule are as follows. First off, it enables parties to work on parallel chains with approximately the same accumulated difficulties at the onset of each epoch, as we shall show that the adversary cannot make much more progress than honest parties on a too big a fraction of the chains. But perhaps more importantly, this new rule renders all attacking strategies that work only on a single chain ineffective. In order for the adversary to compete against any chain in an epoch, either he manages to create a fork in the current epoch (which he will succeed with constant probability), or he has to create forks in previous epochs but on a large fraction of the chains; otherwise, even if the adversary manages to acquire much more difficulty than the honest parties on a small fraction of the chains, by our employment of the $\mathsf{ApproxAgree}$ function, they do not contribute to the difficulty of chains in the next epoch.

Overview of our security analysis. We extend the analytical framework for arguing security on a single PoW chain [GKL17, GKL20] to the parallel-chains setting. Here we highlight the subtleties introduced by the new construction and our adaptations.

Throughout our analysis, we study the m-ary sequences of random variables defined for the m-ary mining procedure, which are conditionally (in)dependent in the following manner. Let E_r denote the execution of the protocol up to round r, and E_r^i the execution up to round r on the i-th chain. First, the random variables at round r corresponding to the i-th parallel chain, defined by the block targets that honest parties will query, are conditioned on E_{r-1}^i. We show that these random variables satisfy martingale properties, thus yielding m-ary parallel martingale sequences. Next, since we bind all the mining procedures using $m\times1$ PoW, in each round, all random variables spanning the m parallel chains are mutually independent. Finally, and more importantly, under our new target recalculation function, the random variable at round r in an epoch is conditioned on the whole execution up to the conclusion of the previous epoch. In other words, the parallel martingale sequences are mutually independent only within the same epoch (as opposed to the previous analyses of parallel blockchains where the m-ary mining procedures are mutually independent throughout the execution [GKS25]).

Taking into account the conditional (in)dependence among the m-ary parallel mining procedures, we now provide a high-level overview of our proof. We begin by defining the properties that we would like to achieve:

1. Honest parties always produce blocks at a steady rate (Equations 10 and 11);
2. in every epoch, a sufficiently large fraction of chains yield freshness and well-boundedness for input-blocks (Equation 12); and
3. at the onset of each epoch, honest parties work on parallel blockchains with approximately the same accumulated difficulty, and their selected mining targets are appropriately bounded with respect to each other (Equations 13, 14 and 15).

Next, we study single-chain convergence probabilities, deriving constant failing probabilities given sequences of random variables of constant length. Further, for any interval where the m-ary mining procedures are mutually independent, we apply the union bound and conclude that, with overwhelming probability, ideal convergence events happen on a sufficiently large fraction of chains.

After all the above preparatory steps, we present an inductive-style argument, which at a very high level proves that, if good properties hold for a given epoch, then they will all hold for the next epoch. Combining with our assumption that the protocol execution begins with a safe genesis block, we obtain that all the desired properties hold throughout the execution of the protocol.

Consensus and SMR with fast difficulty adjustment. On top of our parallel blockchain construction, we can seamlessly run a higher-level protocol (such as consensus, state machine replication [SMR], etc.) using the generic approach presented in [GKS24]. The resulting SMR protocol achieves *expected-constant* settlement for all incoming transactions, and is secure in an environment where the number of participants fluctuates in every window of *constant* duration, yielding:

Theorem (Informal). *Assume that the diffusion network has bounded delay and the fluctuation on number of parties is bounded for every time window of a fixed* constant *number of rounds. Assuming an appropriate parameterization, there exists (i) a consensus protocol that achieves Agreement and Validity and terminates in* expected-constant *time, and (ii) a state machine replication protocol that achieves Consistency and* expected-constant *Liveness, against an adversary that controls a minority of the parties in every round, except with probability negligibly small in the security parameter.*

3 Preliminaries

Throughout the paper, we use κ to denote the security parameter. We write $[n] \triangleq \{1, 2, \ldots, n\}$, and use $\|$ to denote the concatenation of two strings.

Blockchain notation. A block with target $T \in \mathbb{N}$ is a quadruple of the form $\mathcal{B} = \langle ctr, r, h, x\rangle$ where $ctr, r \in \mathbb{N}$, $h \in \{0,1\}^\kappa$ and $x \in \{0,1\}^*$. A blockchain $\mathcal{C}$ is a (possibly empty) sequence of blocks; the rightmost block is denoted by $\mathsf{head}(\mathcal{C})$ ($\mathsf{head}(\varepsilon) = \varepsilon$). These blocks are chained in the sense that if $\mathcal{B}_{i+1} = \langle ctr, r, h, x\rangle$, then $h = H(\mathcal{B}_i)$. We use $\mathsf{TS}(\mathcal{B})$ to denote the *timestamp* of $\mathcal{B}$. We denote by $\mathcal{C}^{\lceil t}$ the chain resulting from "pruning" the least number of rightmost blocks so that all block timestamps are less than t. Let $\mathbb{C} = \langle \mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_m\rangle$ denote m *parallel chains* and $\mathbb{C}_j$ the j-th chain $\mathcal{C}_j$ in $\mathbb{C}$.

For our parallel PoW-blockchain construction, we introduce some basic string notation. For a κ-bit string s, where κ is the security parameter, we will use s_i ($i \in [m]$) to denote the i-th bit of s, $[s]_{i\sim m}$ to denote the i-th segment after s is equally divided into m segments—i.e., $[s]_{i\sim m} = s_{[(i-1)*\kappa/m]+1}, \ldots, s_{i*\kappa/m}$. Further, we will write $[s]^{\mathsf{R}}$ as the reverse of string s (i.e., the string obtained by reversing the order of its bits), and use $[s]^{\mathsf{R}}_{i\sim m}$ to denote the reverse of the i-th segment.

3.1 Protocol Execution Model

We present our protocol based on the computation model for multi-party protocols due to Canetti's formulation of "real world" notion of protocol execution [Can00a, Can00b]. In this model, the environment $\mathcal{Z}$ provides input to parties executing the protocol Π. The adversary $\mathcal{A}$ is a single entity that controls all corrupted parties, which is both "adaptive" in the sense that can take control of parties on the fly, and "rushing" meaning that (in each round) $\mathcal{A}$ can observe the honest parties' actions and then react. We describe the "resources" that may be available to the protocol instances (e.g., access to a "diffusion" channel) as *ideal functionalities* in the terminology of [Can00b].

Clock, random oracle, network and CRS functionalities. We assume parties are equipped with perfectly synchronous clocks, which is modeled as having access to a global functionality $\mathcal{G}_{\mathsf{Clock}}$ [KMTZ13]. Following the conventions in synchronous computation, time in our protocol is divided into discrete intervals called "rounds." The hash function H that is used to generate PoWs is modeled as a random oracle $\mathcal{F}_{\mathsf{RO}}$ (with additionally a "wrapper" functionality to capture access restrictions). Regarding message dissemination, we assume a Δ-bounded delay network [PSs17, GKL24]—i.e., once a new message m is reached to at least one honest party at round r, it is guaranteed that all honest parties receive m no later than $r + \Delta$ (by e.g., honest parties keep echoing messages). The adversary can arbitrarily reorder messages within the Δ-round window. This is captured by a diffusion network functionality $\mathcal{F}_{\mathsf{Diffuse}}$. To eliminate pre-mining attacks, we assume a public setup where the genesis block contains a common reference string with sufficiently high entropy, which we model as $\mathcal{F}_{\mathsf{CRS}}$.

We present a full specification of these functionalities in Appendix B.

3.2 Dynamic Participation

Following the existing convention (e.g., [GKL15]), we assume a "flat" model of computation where each party can make the same number of queries in each round, and where the exact number of parties that are activated in each round is not directly observable by any protocol participant.

To capture dynamic participation, we restrict the environment so that the number of parties will be bounded in a certain fashion. Specifically, we define a "respecting environment" in terms of the number of parties that are activated in each round (cf. [GKL17, GKL20]).

Definition 1 ((γ, s)-respecting sequence). *For $\gamma \in \mathbb{R}^+$ we call the sequence $(n_r)_{r\in[0,B)}$, where $B \in \mathbb{N}$, (γ, s)-*respecting *if for any set $S \subseteq [0, B)$ of at most s consecutive rounds, $\max_{r\in S} n_r \leq \gamma \cdot \min_{r\in S} n_r$.*

Remark 1. For fast difficulty adjustment, the focus of this paper, we consider a **constant** $s \in \mathbb{N}$, while in all previous works $s = \Theta(\mathsf{polylog}\kappa)$. See Table 1.

Remark 2. Looking ahead, in our analysis we will set s slightly larger than the duration of a protocol epoch, thus, in every epoch the number of parties' fluctuation will be bounded by γ.

Next, in order to apply a more fine-grained classification on protocol participants, we identify a special subset of honest parties as *alert* parties, following the treatment in [BGK+18]. Alert parties are a subset of honest parties who have access to all the resources $\mathcal{F}_{\mathsf{RO}}$, $\mathcal{F}_{\mathsf{Diffuse}}$, $\mathcal{G}_{\mathsf{Clock}}$ and are synchronized by participating in the protocol for sufficiently long time, which are the core set of parties to execute the protocol.

Finally, we express our honest-majority assumption, measured by the number of parties that they are allowed per round for alert and non-alert ones (which includes both corrupted and honest-yet-desynchronized parties). We also require that there is a gap between these two types of parties which we denote as δ.

Definition 2 (Honest Majority). *Let h_r, t_r denote the number of alert and non-alert parties at round r respectively. For all $r \in \mathbb{N}$ and $\delta \in (0,1)$, it holds that $h_r > (1-\delta)t_r$.*

3.3 Consensus and State Machine Replication

Finally, we present the definitions of Byzantine agreement (a.k.a. Consensus) and State Machine Replication (a.k.a. Ledger Consensus, Nakamoto Consensus).

Definition 3 (Byzantine Agreement). *A protocol Π solves Byzantine agreement in the synchronous setting provided it satisfies the following two properties:*
- **Agreement:** *There is a round after which all honest parties return the same output;*
- **Validity:** *If all honest parties have the same input, then that should be the output.*

For State Machine Replication, we consider a set of servers, where a subset $\mathcal{H}$ of them is assumed to be non-faulty. The notation $\mathsf{Log}_i[t]$ denotes the log (of timestamped transactions) of the server P_i up to time t. Furthermore, it is assumed that each server has a buffer for incoming transactions, denoted by $I_i[t]$, that are valid with respect to its view.

Definition 4 (State Machine Replication). *A protocol Π solves State Machine Replication provided it satisfies the following two properties:*
- **Consistency:** $\forall \mathsf{P}_i, \mathsf{P}_j \in \mathcal{H}$ *(where not necessarily $i \neq j$) and t, t' it holds that* $\mathsf{Log}_i[t] \preceq \mathsf{Log}_j[t']$ *or* $\mathsf{Log}_j[t'] \preceq \mathsf{Log}_i[t]$*;*
- **Liveness:** *There is a parameter $u \in \mathbb{N}$ for which the following holds:* $(\forall \mathsf{P}_i \in \mathcal{H} : \mathtt{tx} \in I_i[t]) \implies \mathtt{tx} \in \mathsf{Log}_i[t+u]$.[4]

4 PoW Blockchains with Fast Difficulty Adjustment

In this section, we present our new PoW blockchain protocol that can adjust its mining difficulty for epochs of constant duration. We start by describing our parallel-chain framework, followed by our new chain difficulty rule (Section 4.2) and target recalculation function (Section 4.3).

4.1 Our Parallel-Chain Framework

We present an overview of our parallel blockchain framework, highlighting the components that facilitate fast difficulty adjustment.

The mining procedure. As in [GKS24, GKS25], we run m blockchains in parallel. Their mining procedure is bound together using $m\times1$ PoWs — i.e., all the information on m parallel chains is "packed" into one hash query, and the κ-bit output is equally divided into $m = \Theta(\log^2 \kappa)$ "chunks," where each chunk has $\kappa/m = \omega(\log \kappa)$ bits. The i-th chunk is associated with the mining process on the i-th chain, and the hash of a block $\mathcal{B}$ on the i-th chain is accordingly defined as $[H(\mathcal{B})]_{i\sim m}$. Since chunks are non-overlapping and we model the hash function as a random oracle, we achieve full independence among the m mining processes with $m\times1$ PoW.[5]

[4] We allow the liveness parameter u to be a random variable with a distribution that depends on the specific parameterization. In our protocol, we achieve u with a geometric distribution, thus *expected-constant* liveness.

[5] We note that full independence is achieved for m sub-procedures in each RO query. However, the mining procedure of m-ary parallel blockchains are not necessarily mutually independent, as our new target recalculation function in Section 4.3 makes use of information on all parallel chains to decide the next mining target.

In our protocol, RO queries are of the form $\langle ctr, r, h, st, h^*, h', \mathtt{val}\rangle$. We explain each element below. Our protocol also makes use of a second type of blocks which we term "input-blocks"[6]. Input-blocks are used to help adjust the mining difficulty (which we detail in Section 4.3) and provide inputs for the higher-level applications (which we elaborate on in Section 5.4). The mining procedure of input-blocks is bound together with that of regular blocks. As is the case for regular blocks, there are m parallel mining procedures for input-blocks, and the input-blocks that are generated in the i-th procedure can only be included in the i-th chain.

- $ctr \in \{0,1\}^\kappa$: *PoW nonce.* Miners try different nonces every time in order to find a block with a hash value smaller than the mining target.
- $r \in \mathbb{N}$: *Block timestamp.* Blocks on-chain should present monotonically increasing timestamps.
- $h \in \{0,1\}^\kappa$: *Previous-block pointer.* A block $\mathcal{B}$ extends a chain $\mathcal{C}$ if $\mathcal{B}$ points to the last block of $\mathcal{C}$ by including its block hash in $\mathcal{B}$'s block header. With m-ary parallel chains, miners use h to point to m last blocks on each parallel chain, by equally dividing h into m chunks and $[h]_{i\sim m}$ pointing to the last block on the i-th chain in the party's local view.
- $st \in \{0,1\}^\kappa$: *Block states.* The content of blocks, again organized in an m-ary fashion and split into m chunks, one for each parallel chain. Note that chains in our protocol do not include transactions/messages directly; rather, inputs to high-level applications are first packed in input-blocks, and regular blocks include input-blocks only.
- $h^* \in \{0,1\}^\kappa$: *Block references.* The pointers to blocks in previous epochs. In this field, parties insert their local view of the previous epoch, by including the hash of the last block on the i-th chain in the previous epoch in their local view into $[h^*]_{i\sim m}$. h^* will play a central role in our new target recalculation function (Section 4.3).
- $h' \in \{0,1\}^\kappa$: *Fresh randomness.* Input-blocks to be included on the i-th chain need fresh randomness in order to prevent pre-mining attacks (cf. [GKL15, PS17]). Hence, miners provide such fresh randomness for the i-th chain in $[h']_{i\sim m}$.
- $\mathtt{val} \in \{0,1\}^*$: *Input values.* The payload message that a miner can choose to include in the input-block. This field is reserved for the input values of high-level applications, e.g., consensus.

After preparing a block header that contains all the information for the m parallel chains, a miner makes a single random oracle query and gets the output u. When the i-th segment of u is smaller than the mining target T in this epoch (more details on epochs and how T is calculated are presented in the sequel), then $\mathcal{B}$ is a valid block on the i-th chain extending block with hash $[h]_{i\sim m}$. To get a valid input-block with respect to chain index i, miners further check the reversed output chunk $[u]^{\mathsf{R}}_{i\sim m}$, and if $[u]^{\mathsf{R}}_{i\sim m} < T$, then it yields a valid input-block. Notably, $\mathcal{B}$ can be a valid (input-)block on multiple parallel chains.

Due to our m-ary parallel mining design and the adoption of input-blocks, and depending on the type of blocks, certain fields in the output can be dummy for block validation. More specifically, for a block on the i-th chain, fresh randomness h' and input values $\mathtt{val}$ do not play any role in validation, and any fields in h other than $[h]_{i\sim m}$ constitute dummy information, and the same holds for st. Regarding an input-block on the i-th chain, h and st are dummy for validation, and only $[h']_{i\sim m}$ in the h' field matters.

[6] Blocks with similar functionality have appeared in previous works with different names — 'beacons' in [BGK+21, GKS22] and 'input-blocks' in [GKS24, GKS25].

Epochs, input-blocks and blocktrees. We divide the protocol execution time into consecutive, non-overlapping *epochs* of length M rounds. We denote the epoch index by ep starting from 1. Given a block $\mathcal{B}$, the epoch that $\mathcal{B}$ belongs to is $\lceil r/\mathsf{M} \rceil$ where r is the timestamp of $\mathcal{B}$.

An epoch can be further divided into three consecutive non-overlapping stages, which last for $\mathsf{M_{FR}}$, $\mathsf{M_{IB}}$ and $\mathsf{M_{CP}}$, respectively (thus $\mathsf{M} = \mathsf{M_{FR}} + \mathsf{M_{IB}} + \mathsf{M_{CP}}$). In the first stage that lasts for $\mathsf{M_{FR}}$ rounds, parties keep extending the blockchain and wait for the second stage where they can contribute inputs. The last stage of $\mathsf{M_{CP}}$ rounds is used for parties to reach an (oblivious) agreement over their local view of parallel chains in the current epoch.

Our parallel blockchain would use input-blocks in the following way. First, a valid input block w.r.t. the i-th chain $\mathcal{C}$, with timestamp in epoch ep, can only be included in the second stage of ep—in other words, blocks with timestamp $r \in \{(\mathsf{ep}-1)\cdot\mathsf{M} + \mathsf{M_{FR}} + 1, \ldots, \mathsf{ep}\cdot\mathsf{M} - \mathsf{M_{CP}}\}$. In addition, the fresh randomness field of these input-blocks should point to the last block $\mathcal{B}$ on $\mathcal{C}$ with timestamp less than $(\mathsf{ep}-1)\cdot\mathsf{M} + \mathsf{M_{FR}}$—i.e., $[\mathtt{IB}.h']_{i\sim m} = [\mathcal{B}]_{i\sim m}$, thus preventing pre-mining attacks on chains with good properties (i.e., chains that honest parties share a common view at the end of the epoch, which plays a central role in our new difficulty adjustment mechanism).

While honest parties always apply the chain selection rule and work on the heaviest fork in their local view (we will elaborate further on this after introducing our new difficulty rule), in order to adjust the mining difficulty and enable high-level applications, they will need to preserve other possibly non-heaviest chain forks. This extends parallel blockchains to parallel *blocktrees*. The root of this tree is the genesis block and each path corresponds to one fork. Recall that on a single chain, the epoch indices of blocks increase monotonically. Epochs also apply to the blocktrees as every block on the tree must belong to a unique chain.

4.2 A New Chain Difficulty Rule

In this section, we specify how the difficulty of a chain is calculated in our protocol. In contrast to the more conventional approach of counting the accumulated difficulties of all the blocks on the chain, our new chain difficulty rule is defined in a recursive manner across all parallel blockchains.

Before we formally present the new difficulty rule, we introduce additional notation relevant to the new parallel-chain framework. Let $\mathcal{C}^{\mathsf{ep}}$ denote the longest sub-chain of $\mathcal{C}$ where all blocks have timestamps in epoch ep. As we require monotonically increasing timestamps, given $\mathcal{C}$ and ep, $\mathcal{C}^{\mathsf{ep}}$ is unique. Next, let $\mathsf{RefChains}(\mathcal{B})$ denote the set of chains in the previous epoch such that the block hash of their last blocks coincide with the block reference field of $\mathcal{B}$. Formally,

$$\mathsf{RefChains}(\mathcal{B}) \triangleq \left\{ \mathcal{C} \;\middle|\; [H(\mathsf{head}(\mathcal{C}))]_{i\sim m} = [\mathcal{B}.h^*]_{i\sim m} \right\}. \tag{3}$$

Slightly abusing notation, let $\mathsf{RefChains}(\mathcal{C}^{\mathsf{ep}})$ be equal to $\mathsf{RefChains}(\mathcal{B})$ where $\mathcal{B}$ is the left-most block on $\mathcal{C}^{\mathsf{ep}}$ (i.e., the block with minimum block height). Note that, for the first epoch, chains begin with the genesis block, thus $\mathsf{RefChains}(\mathcal{C}) = \emptyset$.

In order to better describe the approximate-agreement sub-component in our protocol, we introduce some array operations following the notation in [DLP+86]. Let $\mathbf{V} = (v_1, \ldots, v_n)$ be a real array of n elements and denote $\tilde{\mathbf{V}} = (\tilde{v}_1, \ldots, \tilde{v}_n)$ the array after ordering $\mathbf{V}$ non-decreasingly. We define two operations reduce and select. Intuitively, reduce with parameter $\mathbf{V}$ and η first orders $\mathbf{V}$ non-decreasingly and then "trims" the η largest and η smallest elements in $\tilde{\mathbf{V}}$; select with parameter $\mathbf{V}$ and η first orders $\mathbf{V}$ non-decreasingly, selects every first element of η consecutive elements, and uses them to form a new array. Formally,

$$\mathsf{reduce}(\mathbf{V}, \eta) = (\tilde{v}_{\eta+1}, \ldots, \tilde{v}_{n-\eta}) \quad \text{and} \quad \mathsf{select}(\mathbf{V}, \eta) = (\tilde{v}_1, \tilde{v}_{\eta+1}, \tilde{v}_{2\eta+1}, \ldots). \tag{4}$$

Additionally, for an array $\mathbf{V}$, let $\mathsf{avg}(\mathbf{V})$ denote the average of its elements, and $\mathsf{med}(\mathbf{V})$ denote the median. In our protocols, these operations are often composed. Thus, for notational brevity, we define

$$\mathsf{ApproxAgree}(\mathbf{V}, \eta) \triangleq \mathsf{avg}(\mathsf{select}(\mathsf{reduce}(\mathbf{V}, \eta), \eta)). \tag{5}$$

I.e., first elements from $\mathbf{V}$ are "trimmed," and then the average of the selected values is computed.

Epoch-based chain difficulty rule. We write $\mathsf{Diff}(\mathcal{B})$ to denote the difficulty of a (valid) block. Let $\mathsf{Diff}(\mathcal{C})$ denote the accumulated difficulty of a chain $\mathcal{C}$. In the first epoch, $\mathsf{Diff}(\mathcal{C})$ is the sum of all blocks' difficulties. Then, starting from the second epoch onwards, let ep denote the epoch index of the last block on $\mathcal{C}$. The difficulty of $\mathcal{C}$ is the sum of two parts: The first part is the sum of difficulties of all blocks in epoch ep, while the second part is derived by applying $\mathsf{ApproxAgree}$ over the vector of chain difficulties of all reference chains of $\mathcal{C}^{\mathsf{ep}}$. We choose μm, where $\mu \in (0, 1)$, as the second parameter in $\mathsf{ApproxAgree}$[7], which controls how the reduce and select operations behave (recall Equation (4)). Formally:

$$\mathsf{Diff}(\mathcal{C}) = \left(\sum_{\mathcal{B} \in \mathcal{C}^{\mathsf{ep}}} \mathsf{Diff}(\mathcal{B}) \right) + \mathsf{ApproxAgree}\Big(\big\{ \mathsf{Diff}(\mathcal{C}') \;\big|\; \mathcal{C}' \in \mathsf{RefChains}(\mathcal{C}^{\mathsf{ep}}) \big\}, \mu m \Big). \tag{6}$$

Given this new rule, whenever a block $\mathcal{B}$ is added to a chain in the current epoch, its accumulated difficulty is increased by $\mathsf{Diff}(\mathcal{B})$. Yet, and in contrast to conventional approaches, when entering a new epoch, chains might yield a lower difficulty compared with their own prefixes in the previous epochs. For example, consider the following scenario: A chain $\mathcal{C}$ in epoch ep has difficulty $\mathsf{Diff}(\mathcal{C})$. After entering $\mathsf{ep} + 1$ and a new block $\mathcal{B}$ is appended to $\mathcal{C}$, it holds that $\mathsf{Diff}(\mathcal{C}\mathcal{B}) < \mathsf{Diff}(\mathcal{C})$ due to the fact that a large fraction of chains in $\mathsf{RefChains}(\mathcal{B})$ has much less accumulated difficulty than $\mathcal{C}$.

We remark that such non-monotonicity on difficulty accumulation is indeed our design intention, and it will not result in difficulty loss across epochs. The reason is that, when entering a new epoch, honest miners extend chains with blocks timestamped in the new epoch only. Further, extending chains in the old epoch does not help, as parties decide mining difficulty and perform state updates for high-level applications using (their view of) the most recent epoch only. We elaborate further on these aspects in the sequel.

Chain validation and selection rules. In our protocol, parties validate incoming chains independently on each chain index, analogously to the process in conventional PoW blockchains (i.e., by checking pointers to previous blocks and block hash value against the mining target), but additionally by performing validity check on block references. I.e., if a block is valid, then m chains in its block reference field h^* must all be valid chains. There are no other checks beyond their existence and validity, and, importantly, these reference chains need not stay on the longest fork in the previous epoch. Further, we also require that blocks on the same chain in the same epoch should not have targets that are τ times larger nor τ times smaller than the first block — this excludes DoS-type attacks by inserting blocks with tiny difficulties (and we shall later prove that such bound does not hurt security as honest parties always approximately agree with the mining target of the first block).

Refer to Algorithm 1 for a code specification of the chain validation procedure. Note that, we will specify how mining target is decided in the next section (in the code target is validated from invoking $\mathsf{ExtractMiningTarget}$). And, since we will check chain validity when extracting mining targets, $\mathsf{IsValidChain}$ validates blocks in the most recent epoch only, as other blocks in previous epochs will be checked in a recursive manner through the $\mathsf{ExtractMiningTarget}$ sub-procedure.

[7] Parameter μ is selected subject to certain conditions; refer to Section 5 for details.

Algorithm $\mathsf{IsValidChain}(\mathcal{C})$

if $\mathcal{C}$ starts with a block other than the genesis $\mathbf{G}$ **then return** false
if $\exists \mathcal{B} \in \mathcal{C}$ s.t. $\mathsf{TS}(\mathcal{B}) > r$ **then return** false ◁ Filter future timestmaps
Let i denote the chain index of $\mathcal{C}$
Compute $ep \leftarrow \lceil \mathsf{TS}(\mathsf{head}(\mathcal{C}))/\mathsf{M} \rceil$
Extract h'_{ep} from $\mathsf{head}(\mathcal{C}^{(ep-1)\cdot \mathsf{M}+\mathsf{M}_{\mathsf{FR}}\rceil}).h'$ ◁ Derive fresh randomness
Let $(\mathcal{B}_u, \ldots, \mathcal{B}_v)$ denote the sequence of blocks on $\mathcal{C}$ in epoch ep
for j **from** u **to** v **do**
    Set $T_j \leftarrow \mathsf{ExtractMiningTarget}(\mathcal{B}_j)$ ◁ Algorithm 3
    if $[H(\mathcal{B}_j)]_{i\sim m} \geq T_j$ **then return** false
    if $[\mathcal{B}_j.h]_{i\sim m} \neq [H(\mathcal{B}_{j-1})]_{i\sim m}$ **or** $\mathsf{TS}(\mathcal{B}_j) \leq \mathsf{TS}(\mathcal{B}_{j-1})$ **then return** false
    if $T_j \geq T_u \cdot \tau$ **or** $T_j \leq T_u/\tau$ **then return** false
    if $(ep-1)\cdot \mathsf{M}_{\mathsf{CP}} + \mathsf{M}_{\mathsf{FR}} < \mathsf{TS}(\mathcal{B}_j) \leq ep \cdot \mathsf{M} - \mathsf{M}_{\mathsf{CP}}$ **then**
        for each input block $\mathtt{IB} \in \mathcal{B}_j$ **do**
            Parse $\mathtt{IB}$ as $\langle ctr', r', \cdot, \cdot, \cdot, h', \cdot \rangle$
            if $r' \leq (ep-1)\cdot \mathsf{M}_{\mathsf{CP}} + \mathsf{M}_{\mathsf{FR}}$ **or** $r' > ep \cdot \mathsf{M} - \mathsf{M}_{\mathsf{CP}}$ **or** $r' \leq r$ **then return** false
            if $\mathcal{C}$ contains duplicate $\mathtt{IB}$ **then return** false
            Set $T_{\mathtt{IB}} \leftarrow \mathsf{ExtractMiningTarget}(\mathtt{IB})$.
            if $[h']_{i\sim m} \neq [h'_{ep}]_{i\sim m}$ **or** $[H(\mathtt{IB})]^{\mathsf{R}}_{i\sim m} \geq T_{\mathtt{IB}}$ **then return** false
            if $T_{\mathtt{IB}} \geq T_u \cdot \tau$ **or** $T_{\mathtt{IB}} \leq T_u/\tau$ **then return** false
        end for
    end if
end for
return true

Algorithm 1: *The chain validation procedure.*

Regarding chain selection, parties choose m-ary parallel chains to mine based on the heaviest-chain selection rule applied independently on each chain index, with "heaviest" defined by our new chain difficulty rule in Equation (6). Note that, since the first block in an epoch decides the $\mathsf{ApproxAgree}$ part of a chain, the adversary can rush to mine the first block $\mathcal{B}$ in epoch ep, thus creating a chain with low difficulty in ep. Nevertheless, honest miners will not switch to such adversarial fork, if the next block that they are to mine can have reference pointers that yield a heavier chain than the adversarial one. The chain validation and selection procedure is presented below. See Algorithm 2 for a full specification of this procedure.

Algorithm $\mathsf{UpdateLocalChain}(\mathbb{C}_{\mathsf{local}}, \mathcal{N} = \{\mathcal{C}_1, \ldots, \mathcal{C}_N\})$

Let r, ep denote the current round and epoch respectively
Let $\mathbb{C}^{ep-1} = (\mathcal{C}_i^{ep-1})_{i\in[n]}$ denote the local chains at the end of epoch $ep-1$
Compute $d = \mathsf{ApproxAgree}((\mathsf{Diff}(\mathcal{C}'))_{\mathcal{C}'\in\mathbb{C}^{ep-1}}, \mu m)$
for i **from** 1 **to** m **do**
    Let $\mathcal{C}_i$ denote the i-th chain in $\mathbb{C}_{\mathsf{local}}$
    Set $\mathcal{C}_{\max} \leftarrow \mathcal{C}_i$ and $d_{\max} \leftarrow \mathsf{Diff}(\mathcal{C}_i)$

if $\nexists \mathcal{B} \in \mathcal{C}_i$ s.t. $\mathsf{TS}(\mathcal{B}) > (ep-1) \cdot \mathsf{M}$ **then** $d_{\max} \leftarrow d$
for each $\mathcal{C}_j \in \mathcal{N}$ with chain index i **do**
if $\mathsf{IsValidChain}(\mathcal{C}_j) = \mathsf{false}$ **then continue**
if $\mathsf{Diff}(\mathcal{C}_j) > d_{\max}$ **then** Set $\mathcal{C}_{\max} \leftarrow \mathcal{C}_j$ and $d_{\max} \leftarrow \mathsf{Diff}(\mathcal{C}_j)$
end for
Replace the i-th chain in $\mathbb{C}_{\mathsf{local}}$ by $\mathcal{C}_{\max}$
end for

Algorithm 2: *Parties filter invalid chains and select the heaviest one.*

On approximate agreement over chain difficulties. Next, we provide some high-level intuition on the design choice of our new chain difficulty rule. Recall that our goal is to secure PoW mining with fast difficulty adjustment in every constant interval; however, previous analyses [GKL17, GKL20] have shown that there is always a constant probability that in constant time bad mining events happen. Hence, we apply ApproxAgree over parallel chains from the previous epoch, thus filtering out chains of undesired high/low difficulties. With foresight, our analysis will show that parties reach approximate agreement on chain difficulties among all parallel chains—i.e., at the onset of each epoch, parties start to mine chains with somewhat similar difficulties, regardless of the potential adversarial mining spikes that might have happened on the same chain index in previous epochs. (For illustration, see the preliminary analysis of ApproxAgree in Lemma 1(a).)

In addition, while chain validation does not require that reference blocks stay on the longest fork of any honest party's local view, we will also prove that the adversary cannot drop too many honest chains as otherwise he will fail to build a chain with competing difficulty compared with the one that honest parties will mine on.

4.3 Difficulty Adjustment in Constant Rounds

In this section we present in detail the novel difficulty adjustment algorithm specially designed for our parallel blockchain construction. Note that the difficulty of a block $\mathsf{Diff}(\mathcal{B})$ is defined as the inverse of its mining target $T(\mathcal{B})$. The same holds for input-blocks (cf. Section 4.1), and we extend the block difficulty/target notation for input-blocks as well.

First, we briefly recall the single-chain target recalculation function proposed in [GKS22], where epochs are of a fixed duration, and the mining target for the next epoch is computed as a product of the block target T in the current epoch and an adjustment rate ρ, defined as the ratio between the actual number of blocks in the current epoch versus its expected number. Our new target recalculation function for parallel blockchains will follow this paradigm, albeit with a different mechanism for choosing targets and ratio in the current epoch. Further, in our protocol parties will *not* reach full agreement over the mining targets, and we allow blocks (and input-blocks) to have different difficulties.

Reference target and adjustment ratio. All blocks in the first epoch adopt the initial mining target T_0. At the beginning of the following epochs, an honest party P computes its own reference target and adjustment ratio that the party will use in the new epoch, based on her local parallel blockchain $\mathbb{C}_{\mathsf{local}}$. We note that, once P selects her own references, P will stick to these values and use them throughout the new epoch.

Regarding the reference target T^{ref}, it is computed by applying $\mathsf{ApproxAgree}$ on the vector of median input-block targets of all reference chains of the last epoch. Recall that, in epoch ep parties mine, generate and include a second type of "input-block" into the blockchain during time interval $\{(\mathsf{ep}-1)\cdot \mathsf{M} + \mathsf{M}_{\mathsf{FR}} + 1, \ldots, \mathsf{ep}\cdot \mathsf{M} - \mathsf{M}_{\mathsf{CP}}\}$. We write $\mathtt{IB} \in \mathcal{C}^{\mathsf{ep}}$ to denote that a (valid) input-block $\mathtt{IB}$ is included in $\mathcal{C}$ during epoch ep; further, we write $\mathtt{IB} \in \mathcal{C}$ to denote that $\mathtt{IB}$ is included in the last epoch of $\mathcal{C}$. When a party P is ready to mine a block $\mathcal{B}$, where she puts her local view of the previous epoch in the block reference, the reference target for $\mathcal{B}$ is formally defined as:

$$T^{\mathsf{ref}}(\mathcal{B}) \triangleq \mathsf{ApproxAgree}\Big(\big\{\mathsf{med}_{\mathtt{IB}\in\mathcal{C}} T(\mathtt{IB}) \;\big|\; \mathcal{C} \in \mathsf{RefChains}(\mathcal{B})\big\}, \mu m\Big), \tag{7}$$

where the same parameter μ is adopted in the chain difficulty rule (Equation (6)).

Analogously, the reference ratio ρ^{ref} is computed by applying $\mathsf{ApproxAgree}$ on the vector of ratios of the expected difficulties versus the actual accumulated ones on each chain. In more detail, in the first epoch all input-blocks are of the same difficulty, hence it suffices to count their numbers. Let Λ denote the ideal number of input-blocks that should appear on the chain. In the first epoch, the reference ratio is thus defined as:

$$\rho^{\mathsf{ref}}(\mathcal{B}) \triangleq \mathsf{ApproxAgree}\big(\{\Lambda/|\{\mathtt{IB} \in \mathcal{C}\}| \;|\; \mathcal{C} \in \mathsf{RefChains}(\mathcal{B})\}, \mu m\big).$$

From the second epoch onwards, since input-blocks might be of different difficulties, simply counting their numbers no longer work. Hence, we define the ratio on each chain with respect to the difficulty of a local reference block on that chain, again selected as the one with the *median* difficulty on the chain. For a chain $\mathcal{C}$, the ideal amount of accumulated difficulty is $\Lambda \cdot \mathsf{med}_{\mathtt{IB}\in\mathcal{C}}\mathsf{Diff}(\mathtt{IB})$, and the actual accumulated difficulty is $\sum_{\mathtt{IB}\in\mathcal{C}} \mathsf{Diff}(\mathtt{IB})$. Thus, the reference ratio of a block $\mathcal{B}$ is formally defined as:

$$\rho^{\mathsf{ref}}(\mathcal{B}) \triangleq \mathsf{ApproxAgree}\left(\left\{\frac{\Lambda \cdot \mathsf{med}_{\mathtt{IB}\in\mathcal{C}}\mathsf{Diff}(\mathtt{IB})}{\sum_{\mathtt{IB}\in\mathcal{C}} \mathsf{Diff}(\mathtt{IB})} \;\middle|\; \mathcal{C} \in \mathsf{RefChains}(\mathcal{B})\right\}, \mu m\right). \tag{8}$$

Looking ahead, in our analysis in Section 5, we will quantify the parameter Λ using the duration of the input-block inclusion phase M_{IB} and the ideal (honest) block generation rate f.[8]

The new target recalculation function. The mining target of block $\mathcal{B}$ is calculated by first identifying the reference target T^{ref} and then adjusting it proportionally to the reference ratio ρ^{ref}. Formally:

$$T(\mathcal{B}) \triangleq T^{\mathsf{ref}}(\mathcal{B}) \cdot \min\big\{\max\big\{\rho^{\mathsf{ref}}(\mathcal{B}), 1/\tau\big\}, \tau\big\}. \tag{9}$$

Note that there is a "dampening factor" τ as an upper/lower bound on the variation that a single target adjustment between consecutive epochs can have. Such bound also exists in Bitcoin's target recalculation function (cf. [GKL17]), and the previous analyses of that function show that it plays a crucial role in securing blockchains with varying computational power as it quashes the raising difficulty attack [Bah13], which aims at creating a private fork by keeping increasing the difficulty and leveraging the high variance of the process. In our function, we also rely on this mechanism to help secure our parallel blockchain from similar types of attacks.

Given our new target recalculation rule, we note that when validating a block $\mathcal{B}$, parties check that the hash of $\mathcal{B}$ is smaller than its own mining target, specified by the reference chains in its block reference field and the target rule in Equations (7) to (9). The target recalculation procedure is shown below.

[8] The respective value of parameter f in Bitcoin is 1 block per 10 minutes.

Algorithm $\mathsf{ExtractMiningTarget}(\mathcal{B})$

Parse $\mathcal{B}$ as $\langle ctr, r, h, st, h^*, h', \mathtt{val}\rangle$
Compute $ep = \lceil r/\mathsf{M} \rceil$
if $ep = 1$ **then return** T_0
Let $(\mathcal{C}_i)_{i\in[m]} = \mathsf{RefChains}(\mathcal{B})$ denote m chains such that $H(\mathsf{head}(\mathcal{C}_i)) = [h^*]_{i\sim m}$
for i **from** 1 to m **do**
    if $\mathsf{IsValidChain}(\mathcal{C}_i) = \mathsf{false}$ **then abort**
    Extract $IB = \{\mathtt{IB} \in \mathcal{B}' \in \mathcal{C}_i \mid (ep-1)\cdot \mathsf{M} + \mathsf{M_{FR}} < \mathsf{TS}(\mathcal{B})' \leq ep \cdot \mathsf{M} - \mathsf{M_{CP}}\}$
    Compute $T_i = 1/\mathsf{med}_{\mathtt{IB}\in IB}\mathsf{Diff}(\mathtt{IB})$
    Compute $\rho_i = \mathsf{med}_{\mathtt{IB}\in IB}\mathsf{Diff}(\mathtt{IB}) \cdot \Lambda / \sum_{\mathtt{IB}\in IB} \mathsf{Diff}(\mathtt{IB})$
end for
Compute $T = \mathsf{ApproxAgree}((T_i)_{i\in[m]}, \mu m)$
Compute $\rho = \mathsf{ApproxAgree}((\rho_i)_{i\in[m]}, \mu m)$
return $T_{\mathcal{B}} = T \cdot \rho$

Algorithm 3: *Extracting the mining target for a block* $\mathcal{B}$.

On approximate agreement over mining targets. The high-level intuition behind our new target recalculation function is that in each epoch, honest parties reach approximate agreement over their reference targets and reference ratios, thus approximately agreeing on their mining targets.

In more detail, consider reference targets; we will prove that on a sufficiently large fraction of the m parallel chains, honest parties contribute input-blocks that account for the majority of total input-block difficulties. Thus, the median rule guarantees that parties pick a block which, while not necessarily honestly generated, yields a block difficulty that is bounded by the maximum and minimum honest targets from the previous epoch. Making use of our preliminary analysis on $\mathsf{ApproxAgree}$ (see Lemma 1(b)), we show that honest parties possess reference targets that are close to each other. Analogously, we show that adjustment ratios on a large fraction of the chains will be bounded, thanks to the bound on median input-block difficulty and the good convergence bounds on honestly accumulated difficulties. This holds even if the adversary chooses to join forces with honest parties on selected sub-chains and completely drop out from mining (by not publishing new blocks) on the others.

In order to further illustrate the security of our new target recalculation function, let us re-evaluate the raising difficulty attack [Bah13], where the adversary, in the context of parallel blockchains, tries to mine private chains with high difficulty, thus leveraging the high variance to hope for a possible "spark" and gain more difficulty than the honest parties. Our protocol construction is resilient to this type of attack, even if the duration of the epochs is reduced from polylogarithmic to constant, since in order to launch it the adversary needs to prepare blocks with high difficulty on *a large fraction* of the parallel chains. As our analysis shows, while the adversary can produce a large number of heavy blocks on a small fraction of the chains, all these chains will be filtered out by our use of $\mathsf{ApproxAgree}$.

Finally, in our new function we do not require that blocks in the same epoch share the same mining target. Thus, the adversary cannot rush to mine the first block in an epoch and force honest parties to follow his selected block references. This is a natural relaxation as parties will not be able to reach full agreement within constant time. Furthermore, as all block targets have to be validated by block references, the adversary will not have the power to build arbitrarily high

difficulty.

5 Security Analysis

Since our protocol applies the $\mathsf{ApproxAgree}$ function (Equation (5)) on parallel chains, we first establish a lemma on the concentration quality of this function. The first part in Lemma 1 aims at bounding the absolute difference among outputs, as prescribed by standard approximate agreement; the second part, however, provides bounds on the *ratio* between the maximum and minimum output, which plays a crucial role in our analysis and which, to our knowledge, has not been explored in "classical" approximate agreement applications.

Lemma 1. *Let $\mu \in (0, 1/3)$ and $\Phi \in \mathbb{R}^+$. Let $\mathbf{U} = \{u_i\}_i^m$ and $\mathbf{V} = \{v_i\}_i^m$ be two sets of m integers each, and let $\mathbf{W} = \mathbf{U} \cap \mathbf{V}$ denote their intersection.*
(a) If $|\mathbf{W}| > (1-\mu)m$ and $\max \mathbf{W} - \min \mathbf{W} \leq \Phi$, then

$$\left|\mathsf{ApproxAgree}(\mathbf{U}, \mu m) - \mathsf{ApproxAgree}(\mathbf{V}, \mu m)\right| \leq \frac{\mu}{1-\mu}\,\Phi.$$

(b) If $|\mathbf{W}| > (1-\mu)m$ and $\max \mathbf{W} / \min \mathbf{W} \leq \Phi$, then

$$\left|\mathsf{ApproxAgree}(\mathbf{U}, \mu m) / \mathsf{ApproxAgree}(\mathbf{V}, \mu m)\right| \leq 1 + \frac{\mu}{1-\mu}\,(\Phi - 1).$$

Proof. For part (a), we assume w.l.o.g. that $\mathsf{ApproxAgree}(\mathbf{U}, \mu m) > \mathsf{ApproxAgree}(\mathbf{V}, \mu m)$ and $|w| \bmod \mu m = 1$. Consider a new set $\mathbf{U}' = \mathbf{W} \cup \{+\infty\}^{m-|\mathbf{W}|}$; it holds that for any $\mathbf{U}$, $\mathsf{ApproxAgree}(\mathbf{U}', \mu m) \geq \mathsf{ApproxAgree}(\mathbf{U}, \mu m)$. Similarly, for a new set $\mathbf{V}' = \mathbf{W} \cup \{-\infty\}^{m-|\mathbf{W}|}$, it holds that for any $\mathbf{V}$, $\mathsf{ApproxAgree}(\mathbf{V}', \mu m) \leq \mathsf{ApproxAgree}(\mathbf{V}, \mu m)$. Thus, it holds that

$$\begin{aligned}
\left|\mathsf{ApproxAgree}(\mathbf{U}, \mu m) - \mathsf{ApproxAgree}(\mathbf{V}, \mu m)\right| &\leq \mathsf{ApproxAgree}(\mathbf{U}', \mu m) - \mathsf{ApproxAgree}(\mathbf{V}', \mu m) \\
&= \mathsf{avg}(w_{\mu m+1}, w_{2\mu m+1}, \ldots, w_{|\mathbf{W}|}) - \mathsf{avg}(w_1, w_{\mu m+1}, \ldots, w_{|\mathbf{W}|-\mu m}) \\
&\leq \frac{w_{|\mathbf{W}|} - w_1}{(1-\mu)m/\mu m} \leq \frac{\mu}{1-\mu}\,\Phi,
\end{aligned}$$

where $\mathbf{W} = \{w_1, \ldots w_{|\mathbf{W}|}\}$ is ordered non-decreasingly.

For part (b), we similarly define $\mathbf{U}'$ and $\mathbf{V}'$. Then, it holds that

$$\begin{aligned}
\left|\mathsf{ApproxAgree}(\mathbf{U}, \mu m) / \mathsf{ApproxAgree}(\mathbf{V}, \mu m)\right| &\leq \mathsf{ApproxAgree}(\mathbf{U}', \mu m) / \mathsf{ApproxAgree}(\mathbf{V}', \mu m) \\
&= \mathsf{avg}(w_{\mu m+1}, w_{2\mu m+1}, \ldots, w_{|\mathbf{W}|}) / \mathsf{avg}(w_1, w_{\mu m+1}, \ldots, w_{|\mathbf{W}|-\mu m}) \\
&\leq \frac{w_{\mu m+1} + \ldots + w_{|\mathbf{W}|-\mu m} + \Phi \cdot w_1}{w_1 + w_{\mu m+1} + \ldots + w_{|\mathbf{W}|-\mu m}} \leq 1 + \frac{\mu}{1-\mu}\,(\Phi - 1),
\end{aligned}$$

where $\mathbf{W} = \{w_1, \ldots w_{|\mathbf{W}|}\}$ is ordered non-decreasingly. □

In our analysis we apply a variety of large deviation bounds, including Chernoff bounds (Theorem 20) and convergence bounds for martingales (Theorem 21); see Appendix A.

5.1 Additional Notation and Preliminary Results

Our probability space is over all protocol executions of length at most $\mathsf{poly}(\kappa)$ (recall that κ is the security parameter); we will use $\mathbf{Pr}$ to denote the probability measure of this space. Let $\mathcal{E}$ be a

random variable taking values on this space and with a distribution induced by the random coins of all entities (adversary, environment, parties) and the random oracle.

Recall that in our protocol parties mine under the same target for all parallel chains. On any chain indexed $i \in [m]$, if at round r exactly h (alert) parties query the oracle with target T, the probability that at least one of them will succeed is

$$f(T,h) = 1-(1-pT)^h \leq pTh, \text{ where } p = 1/2^{\kappa/m}.$$

At round r, alert parties might be querying the random oracle for various targets. We denote by $T_r^{\min}$ and $T_r^{\max}$ the minimum and maximum of those targets.

Ideal block generation rate. The initial target T_0 implies an initial estimate of the number of honest RO queries h_0; specifically, $h_0 = 2^{\kappa/m}\Lambda_{\mathsf{epoch}}/(T_0 \cdot \mathsf{M}_{\mathsf{IB}})$—i.e., the number of parties it takes to produce Λ_{epoch} blocks of difficulty $1/T_0$ in time M_{IB}. For convenience, we will denote $f_0 \triangleq f(T_0, h_0)$ and simply refer to it as f. Also note that the ideal number of blocks $\Lambda_{\mathsf{epoch}} = \mathsf{M}_{\mathsf{IB}} \cdot f$, so in the analysis we will use $\mathsf{M}_{\mathsf{IB}} \cdot f$ to represent Λ_{epoch}.

Ideal properties and predicates. We will let $\mathcal{S}_r^i$ denote the *set of chains of interest* at round r on chain index i, which will include: (i) Any chain $\mathcal{C}$ that belongs to an alert party at round r; (ii) any chain $\mathcal{C}$ such that $\mathsf{Diff}(\mathcal{C}) > \mathsf{Diff}(\mathcal{C}')$ where $\mathcal{C}'$ belongs to an alert party; and (iii) any chain $\mathcal{C}$ such that $\mathsf{Diff}(\mathcal{C}) = \mathsf{Diff}(\mathcal{C}')$ where $\mathcal{C}'$ belongs to an alert party and $\mathsf{head}(\mathcal{C})$ was computed no later than $\mathsf{head}(\mathcal{C}')$. For an epoch ep, we also write $\mathcal{S}_{\mathsf{ep}}^i$ to refer to $\mathcal{S}_r^i$ where $r = \mathsf{ep} \cdot \mathsf{M}$—i.e., the chain set w.r.t. the last round in ep.

Next, we present some definitions about chains and epochs which will allow us to introduce a few "good" properties, serving as an intermediate step towards proving the desired final properties.

- A chain is *stale* for some round u if it does not contain an honest block computed after round $u - \ell - 2\Delta$.
- An epoch ep is *good* if (i) for round $u = (\mathsf{ep}-1)\cdot\mathsf{M}+1$, it holds that

$$f/2\gamma \leq ph_u T_u^{\min} \leq ph_u T_u^{\max} \leq (1+\delta)\gamma f; \tag{10}$$

  and (ii) for any round $r \in \{(\mathsf{ep}-1)\cdot\mathsf{M}+1, \ldots, \mathsf{ep}\cdot\mathsf{M}\}$, it holds that

$$f/2\gamma^2 \leq ph_r T_r^{\min} \leq ph_r T_r^{\max} \leq (1+\delta)\gamma^2 f. \tag{11}$$

- A chain is *good* for some epoch ep if (i) the majority (counted in terms of difficulty) of the input-blocks are produced by honest parties; and (ii) the ratio between the total input-block difficulty and their median is well-bounded. Formally:

$$\frac{1}{2(1+\delta)\gamma^2} \cdot \mathsf{M}_{\mathsf{IB}} \cdot f \leq \frac{\sum_{\mathtt{IB}\in\mathcal{C}^{\mathsf{ep}}} \mathsf{Diff}(\mathtt{IB})}{\mathsf{med}_{\mathtt{IB}\in\mathcal{C}^{\mathsf{ep}}}\mathsf{Diff}(\mathtt{IB})} \leq 2(1+\delta)\gamma^2 \cdot \mathsf{M}_{\mathsf{IB}} \cdot f. \tag{12}$$

- An epoch ep is *concentrated*[9] if

  (a) for round $u = (\mathsf{ep}-1)\cdot\mathsf{M}+1$ and $U = \{u, \ldots, \mathsf{ep}\cdot\mathsf{M}\}$, it holds that

$$\max\{\mathsf{Diff}(\mathcal{C}) \mid \mathcal{C} \in \mathcal{S}_u^i \in \mathcal{S}_u\} - \min\{\mathsf{Diff}(\mathcal{C}) \mid \mathcal{C} \in \mathcal{S}_u^i \in \mathcal{S}_u\} \leq 3\mu \cdot ph(U); \tag{13}$$

[9] This property establishes the bound on honest parties' views on chain difficulties, mining targets and adjustment ratios; "concentrated" refers to the fact that they are well-bounded with respect to a protocol parameter μ—the upper bound on the fraction of chains that parties do not share a consistent view.

(b) for any round $r \in \{(\mathsf{ep}-1)\cdot \mathsf{M}+1,\ldots,\mathsf{ep}\cdot \mathsf{M}\}$, it holds that

$$T_r^{\min}\cdot(1+2\mu)\geq T_r^{\max}; \tag{14}$$

(c) for the set $\mathcal{H}^{\mathsf{ep}}$ of all alert parties in epoch ep, let $\rho_{\mathsf{P}}^{\mathsf{ref}}$ denote the reference ratio of $\mathsf{P}\in\mathcal{H}^{\mathsf{ep}}$ for mining the next block according to Equation (8); it holds that

$$\min_{\mathsf{P}\in\mathcal{H}^{\mathsf{ep}}}\rho_{\mathsf{P}}^{\mathsf{ref}}\cdot(1+\mu)\geq\max_{\mathsf{P}\in\mathcal{H}^{\mathsf{ep}}}\rho_{\mathsf{P}}^{\mathsf{ref}}. \tag{15}$$

Next, we define a series of useful predicates with respect to $\mathcal{S}_r^i$ the chain set of interest at round r. Let:

- NoStaleChains$(i,\mathsf{ep})\triangleq$ "There is no stale chains in $\mathcal{S}_r^i$, for any $r\in\{(\mathsf{ep}-1)\cdot\mathsf{M}+\mathsf{M}_{\mathsf{FR}}+1,\ldots,\mathsf{ep}\cdot\mathsf{M}\}$."[10]
- NoStaleChains$(\mathsf{ep})\triangleq$ "There is a subset $S\subseteq[m]$ with size $|S|>(1-\mu)\cdot m$ such that for all $i\in S$, NoStaleChains(i,ep) holds."
- GoodEpochs$(\mathsf{ep})\triangleq$ "All epochs $\mathsf{ep}'\leq\mathsf{ep}$ are good."
- GoodChains$(\mathsf{ep})\triangleq$ "There is a subset $S\subseteq[m]$ with size $|S|>(1-\mu)\cdot m$ such that for all $i\in S$, all chains in $S^i_{\mathsf{ep}\cdot M}$ are good."
- ConcentratedEpochs$(\mathsf{ep})\triangleq$ "All epochs $\mathsf{ep}'\leq\mathsf{ep}$ are concentrated."

Random variables. For the purpose of estimating the difficulty acquired by alert parties during a time window on each chain, we define the following random variables w.r.t. round r and chain index i.

- D_r^i ($\overline{D}_r^i$ resp.): The sum of the difficulties of all blocks (resp., input-blocks) computed by alert parties at round r on the i-th chain.
- Y_r^i: The maximum difficulty of all blocks computed by alert parties at round r on the i-th chain.
- Q_r^i: Equal to Y_r^i when $D_u^i=0$ for all $r<u<r+\Delta$, and 0 otherwise.

We call a round r such that $D_r^i>0$ *successful* w.r.t. the i-th chain, and one where $Q_r>0$ *isolated successful* w.r.t. the i-th chain. An isolated successful round guarantees the irreversible progress of the honest parties on the corresponding chain. We also note that isolated success is defined by not having any honest success in the future Δ rounds.

Recall that the total number of RO queries alert parties (resp., the adversary) can make at round r is denoted by h_r (resp., t_r). For a sequence of rounds U, we write $h(U)=\sum_{r\in U}h_r$, and similarly $t(U),D^i(U),Q^i(U),\overline{D}^i(U)$, where the superscript indicates the corresponding chain index.

Regarding the adversary $\mathcal{A}$, while $\mathcal{A}$ may query the random oracle for an arbitrarily low target value and obtain blocks with arbitrarily high difficulty, we wish to upper-bound the difficulty it can accrue during a sequence of J queries. Consider a sequence of consecutive adversarial queries J, associate with it the target of the first query, and denote it by $T(J)$.[11] We define $A^i(J)$ and $B^i(J)$ ($\overline{A}^i(J)$ and $\overline{B}^i(J)$) to be equal to the sum of the difficulties of all blocks (input-blocks resp.) computed by the adversary on chain i during queries in J for target at least $T(J)/\tau$ and $T(J)$, respectively. That is, queries in J for targets less than $T(J)/\tau$ (resp. $T(J)$) do not contribute to $A^i(J)$ (resp. $B^i(J)$).

[10] We define NoStaleChains excluding the first M_{FR} rounds in an epoch. This is because the adversary, when holding chains with irregularly high difficulties in the previous epoch (recall that he can manage to do so on a small fraction of chains), can privately mine an fork exploiting such advantage and thus being slightly ahead of the honest parties for at most M_{FR} rounds.

[11] Recall that in our protocol a query sets the same target for all parallel chains.

Let $\mathcal{E}_{r-1}$ fix the execution just before time r. In particular, a value E_{r-1} of $\mathcal{E}_{r-1}$ determines the adversarial strategy and so determines the targets against which every party will query the oracle at time r and the number of parties h_r and t_r, but it does not determine D_r or Q_r. For an adversarial query j we will write E_{j-1} to refer to the execution just before this query. The following lemma was proven in [GKL20]:

Lemma 2 ([GKL20]). *Let U be a (γ, s)-respecting sequence (Definition 1) such that $|U| \geq s$ and $S \subseteq U$. Then, it holds that*

$$(a)\ \forall h \in \{h_r : r \in U\}, \frac{h}{\gamma} \leq \frac{h(S)}{|S|} \leq \gamma h \text{ and } (b)\ h(U) \leq \big(1 + \frac{\gamma|U \setminus S|}{|S|}\big)h(S).$$

Bad events on the random oracle. We consider three bad events with respect to the random oracle, namely, block insertions, copies and predictions. An *insertion* happens when a block $\mathcal{B}^*$, created after two consecutive blocks $\mathcal{B}$ and $\mathcal{B}'$ on chain $\mathcal{C}$, yields $\mathcal{B}, \mathcal{B}^*, \mathcal{B}'$ three consecutive blocks of a valid chain; a *copy* occurs if the same block exists in two different positions; a *prediction* occurs when a block extends one with later creation time.

Note that, with parallel chains, these bad events should be considered with respect to each of the chains. For example, an insertion with respect to chain index $i \in [m]$ happens if the i-th segment of hash of block $\mathcal{B}^*$ is identical to the i-th segment of hash of block $\mathcal{B}$, implying a "partial" collision on two different RO queries. We show in the following theorem that for all parallel chains, bad events happen with only negligible probability.

Lemma 3. *Consider an execution E of $L = \mathsf{poly}(\kappa)$ time steps. No insertions, no copies, and no predictions occurred in E, except with probability negligibly small in κ.*

Chain growth lemma. Our next step is to show that honest successes as defined by random variables D, Q can indeed contribute to the accumulated difficulties on-chain (i.e., chain growth [GKL17]). We consider a restricted form of the chain-growth lemma, which provides bounds on honestly accumulated difficulties in the same epoch.

Lemma 4 (Same-Epoch Chain Growth). *Let u, v be two rounds of an execution E in epoch ep. If at time u an honest party diffuses a chain of difficulty d with chain index i, then by time v every honest party has received a chain with chain index i of difficulty at least $d + Q^i(S)$, where $S = \{r : u + \Delta \leq r \leq v - \Delta\}$.*

Protocol parameters. In order to argue for ideal convergence probabilities, we split a protocol execution into sequences of consecutive r of rounds. We let ℓ denote the minimum length of the interval that we shall be concerned with:

$$\ell \triangleq \Theta\Big(\frac{4(1+5\epsilon)}{\epsilon^2 f[1-(1+\delta)\gamma^2 f]^{\Delta+1}} \cdot \gamma^3 \cdot \max\{\Delta, \tau\}\Big). \tag{16}$$

Note that ℓ is a constant and is chosen independently of the security parameter κ. Next, we define the duration of an epoch M as well as three underlying stages based on ℓ. Again, note that they are all constants. (See Condition (C3) for a bound on these parameters.)

$$\mathsf{M}_{\mathsf{IB}} \triangleq \mathsf{M}_{\mathsf{CP}} \triangleq \frac{8\gamma}{\epsilon} \cdot (\ell + 2\Delta) \text{ and } \mathsf{M}_{\mathsf{FR}} \triangleq 2(\mathsf{M}_{\mathsf{IB}} + \mathsf{M}_{\mathsf{CP}}). \tag{17}$$

To make the arguments go through, we will assume the following conditions on protocol parameters: First, the advantage of the honest parties δ is large enough to absorb errors from the convergence quality ϵ. Specifically,

$$\epsilon = \mu/3 \leq \delta/24 \leq 1/24 \text{ and } \gamma \leq 1 + \mu. \tag{C1}$$

Next, the network delay Δ and party fluctuation ratio γ are well set with respect to ϵ:

$$[1 - (1 + \delta)\gamma^2 f]^{\Delta} \geq 1 - \epsilon. \tag{C2}$$

5.2 Ideal Convergence Events

Next, we define a few ideal events for the convergence of sequences of random variables.

Definition 5 (Ideal-convergence events). *We define the following ideal-convergence events with respect to a chain index $i \in [m]$:*

- *For a set U of consecutive rounds, let $\mathsf{goodConv}^i_{\mathcal{H}}(U)$ denote the event that $Q^i(U)$ and $D^i(U)$ stay close to their expected value.*

$$\mathsf{goodConv}^i_{\mathcal{H}}(U) \triangleq \begin{pmatrix} (1-\epsilon)[1-(1+\delta)\gamma^2 f]^{\Delta} ph(U) < Q^i(U) \\ \wedge\ D^i(U) < (1+\epsilon)ph(U) \\ \wedge\ (1-\epsilon)ph(U) < \overline{D}^i(U) < (1+\epsilon)ph(U). \end{pmatrix}.$$

- *For a set J of consecutive adversarial queries and*

$$\alpha(J,k) = \frac{\epsilon f[1-(1+\delta)\gamma^2 f]^{\Delta}}{2(1+4\epsilon)\gamma^3\tau} \cdot \frac{k}{T(J)}, \tag{18}$$

let $\mathsf{goodConv}_{\mathcal{A}}(J,k)$ denote the event that $A^i(J)$ and $B^i(J)$ are well upper-bounded.

$$\mathsf{goodConv}^i_{\mathcal{A}}(J,k) \triangleq \begin{pmatrix} A^i(J) < p|J| + \max\{\epsilon p|J|, \tau\alpha(J,k)\} \\ \wedge\ B^i(J) < p|J| + \max\{\epsilon p|J|, \alpha(J,k)\} \\ \wedge\ \overline{A}^i(J) < p|J| + \max\{\epsilon p|J|, \tau\alpha(J,k)\} \\ \wedge\ \overline{B}^i(J) < p|J| + \max\{\epsilon p|J|, \alpha(J,k)\} \end{pmatrix}.$$

- *For a set U of consecutive rounds in a good epoch, let $\mathsf{goodConv}^i(U,k)$ denote the event where, for any subsequence U' of U such that $|U'| \geq k$, $\mathsf{goodConv}^i_{\mathcal{H}}(U')$ and $\mathsf{goodConv}_{\mathcal{A}}(U',k)$ hold. I.e.,*

$$\mathsf{goodConv}^i(U,k) \triangleq \bigwedge_{|U'|\geq k} \Big(\mathsf{goodConv}^i_{\mathcal{H}}(U') \wedge \mathsf{goodConv}^i_{\mathcal{A}}(J(U'),k)\Big),$$

where $J(U')$ is the set of adversarial queries in U'.

Given the above events, we next establish their error probabilities considering sequences consisting of constantly many random variables. Large convergence bounds show that the error probabilities decrease exponentially with respect to the length of the sequences.

Lemma 5. *In a (γ, s)-respecting environment, for any chain $\mathcal{C}_i$ and any set U of at least $k \geq \ell$ consecutive rounds in a good epoch, $\mathsf{goodConv}^i_{\mathcal{H}}(U)$ holds except with probability no more than $\epsilon_{\mathcal{H}}(k)$, where*

$$\epsilon_{\mathcal{H}}(k) \triangleq \exp\left\{ \ln(\Delta+2) - \frac{\epsilon^2 f[1-(1+\delta)\gamma^2 f]^{\Delta+1}}{4\gamma^3\Delta(1+\epsilon/3)} \cdot \min\{k, s\} \right\}.$$

Proof. Fix a chain index and drop all related superscripts on random variables. Fix an execution E_0 just before the beginning of S.

We first consider the lower bound on $Q(S)$. For each nominal time $i \in S$, define a Boolean random variable F_i equal to 1 exactly when all h_i queries of the honest parties yield evaluations above $\min\{T : f(T, h_i) \geq (1+\delta)\gamma^2 f\}$; define $Z_i = Y_i \cdot F_{i+1} \cdots F_{i+\Delta-1}$. Let G denote the event that the time steps in S are good. Given G, for any $i \in S, (F_i = 1) \implies (D_i = 0)$ and so $Q_i \geq Z_i$. For any d, it holds that

$$\mathbf{Pr}\Big[G \wedge \sum_{i\in[k]} Q_i \leq d\Big] \leq \mathbf{Pr}\Big[G \wedge \sum_{i\in[k]} Z_i \leq d\Big]$$

thus we now work on Z_i.

Identify S with $\{1, \ldots, |S|\}$ and partition it with sets of the form $S_j = \{j, j+\Delta, j+2\Delta, \ldots\}$ for $j \in \{0, 1, \ldots, \Delta-1\}$. Fix a set $S_j = \{s_1, s_2, \ldots, s_\nu\}$, with $\nu \geq \lceil |S|/\Delta \rceil$, and define the event G_t as the conjunction of the events G and $t = \epsilon(1-2\gamma^2 f)^\Delta p h(S_j)$. We consider the following event:

$$G_t \wedge \sum_{i\in S_j} Z_i \leq [1-(1+\delta)\gamma^2 f]^\Delta p \sum_{i\in S_j} h_i - t.$$

To that end, consider the sequence of random variables

$$X_0 = 0; X_u = \sum_{i\in[u]} Z_{s_i} - \sum_{i\in[u]} \mathbb{E}[Z_{s_i}|\mathcal{E}_{s_i-1}], u \in [\nu].$$

This is a martingale with respect to the sequence $\mathcal{E}_{s_1-1}(\mathcal{E}_0 = E_0), \ldots, \mathcal{E}_{s_\nu-1}, \mathcal{E}$ because, following the linearity of conditional expectation and the fact that X_{u-1} is a deterministic function of $\mathcal{E}_{s_{u-1}+\Delta-1} = \mathcal{E}_{s_u-1}$, it holds that

$$\mathbb{E}[X_u|\mathcal{E}_{s_u-1}] = \mathbb{E}\big[Z_{s_u} - \mathbb{E}[Z_{s_u}|\mathcal{E}_{s_u-1}]\big|\mathcal{E}_{s_u-1}\big] + \mathbb{E}[X_{u-1}|\mathcal{E}_{s_u-1}] = X_{u-1}.$$

In addition, given an execution E satisfying G_t,

$$\epsilon \sum_{i\in S_j} \mathbb{E}[Z_i|\mathcal{E}_{s_u-1} = E_{s_u-1}] \geq \epsilon \sum_{i\in S_j} [1-(1+\delta)\gamma^2 f]^\Delta p h_i = t.$$

Now, consider the details relevant to Theorem 21. For an execution E satisfying G_t, let B denote the event $\mathcal{E}_{s_u-1} = E_{s_u-1}$. Note that $Z^2_{s_u} = Y^2_{s_u} \cdot F_{s_u+1} \cdots F_{s_u+\Delta-1}$ and all these random variables are independent given B. Since $X_u - X_{u-1} = Z_{s_u} - \mathbb{E}[Z_{s_u}|\mathcal{E}_{s_u-1}]$, let $S^{(u)}_j$ denote a sub sequence of S_j of length $\min\{\nu, s/\Delta\}$ such that $s_u \in S^{(u)}_j$. It holds that

$$\begin{aligned} Z_{s_u} - \mathbb{E}[Z_{s_u}|B] &\leq \frac{1}{T^{\min}_{s_u}} \leq \frac{p h_{s_u}}{p h_{s_u} T^{\min}_{s_u}} \\ &\leq \frac{\gamma p h(S^{(u)}_j)}{p h_{s_u} T^{\min}_{s_u} |S^{(u)}_j|} \leq \frac{2\gamma^3 t}{\epsilon(1-2\gamma^2 f)^\Delta f \cdot \min\{\nu, s/\Delta\}} \stackrel{\text{def}}{=} b. \end{aligned} \tag{19}$$

The third inequality holds due to Lemma 2(a); and the next one is because s_u is a good time step. We see that the event G implies $X_u - X_{u-1} \le b$.

With respect to $V = \sum_u \mathbf{Var}[X_u - X_{u-1}|\mathcal{E}_{s_u-1}] \le \sum_u \mathbb{E}[Z_{s_u}^2|\mathcal{E}_{s_u-1}]$, first recall Lemma 2(a), we have

$$\sum_{u\in[\nu]} \big(ph_{s_u}\big)^2 \le \sum_{u\in[\nu]} ph_{s_u} \frac{p\gamma h(S_j^{(u)})}{|S_j^{(u)}|} \le \frac{\gamma}{\min\{\nu, s/\Delta\}} \big(ph(S_j)\big)^2.$$

Then, using the independence of the random variables it holds that

$$\begin{aligned}
\sum_{u\in[\nu]} \mathbb{E}[Z_{s_u-1}^2|B] &\le [1-(1+\delta)\gamma^2 f]^{\Delta-1} \sum_{u\in[\nu]} \frac{\big(ph_{s_u}\big)^2}{ph_{s_u} T_{s_u}^{\min}} \qquad (20)\\
&\le \frac{2\gamma^3[1-(1+\delta)\gamma^2 f]^{\Delta-1}}{f\cdot\min\{\nu, s/\Delta\}} \cdot \big(ph(S_j)\big)^2 \\
&\le \frac{2\gamma^3 t^2}{\epsilon^2 f(1-2\gamma^2 f)^{\Delta+1}\cdot\min\{\nu, s/\Delta\}} \overset{\text{def}}{=} v. \qquad (21)
\end{aligned}$$

The first inequality holds due to the fact that $\mathbb{E}[(Y_i)^2|\mathcal{E}_{r-1} = E_{r-1}] \le ph_r/T_r^{i,\min}$; the second one is because the inequality above and that all time steps in S_j are good; and the last one is acquired by substituting t.

After applying Theorem 21, we have

$$\begin{aligned}
\mathbf{Pr}[-X_\nu \ge t \wedge G_t] &\le \exp\Big\{-\frac{t^2}{2v(1+\epsilon/3)}\Big\} \\
&\le \exp\Big\{-\frac{\epsilon^2 f[1-(1+\delta)\gamma^2 f]^{\Delta+1}}{4\gamma^3\Delta(1+\epsilon/3)}\cdot\min\{k,s\}\Big\}.
\end{aligned}$$

Note that the first inequality follows $bt < \epsilon v$; and the next one holds by substituting b, v as in Equations (19) and (21) and $\nu \le k/\Delta$. Finally, we apply the union probability to all $j \in \{0, 1, \ldots, \Delta-1\}$ thus

$$\begin{aligned}
&\mathbf{Pr}\Big[G \wedge \sum_{i\in[k]} Q_i \le (1-\epsilon)(1+\delta\gamma^2 f)^\Delta ph(S)\Big] \\
&\quad\le \exp\Big\{\ln\Delta - \frac{\epsilon^2 f[1-(1+\delta)\gamma^2 f]^{\Delta+1}}{4\gamma^3\Delta(1+\epsilon/3)}\cdot\min\{k,s\}\Big\} \overset{\text{def}}{=} \epsilon_Q(k).
\end{aligned}$$

Regarding the bounds on $D(S)$, we consider per honest query. Let J denote the queries in S ($\nu = |J|$), and Z_i the difficulty of any block obtained from query $i \in J$. For the lower bound, define the martingale sequence

$$X_0 = 0; X_u = \sum_{i\in[u]} Z_i - \sum_{i\in[u]} \mathbb{E}[Z_i|\mathcal{E}_{i-1}], u \in [\nu]$$

and let $t = \epsilon p\nu$. Analogously, by considering a sub sequence of S of length $\min\{\nu, s\}$, we have

$$X_u - X_{u-1} \le \frac{2\gamma^3 t}{\epsilon f\cdot\min\{\nu,s\}} \overset{\text{def}}{=} b \text{ and } V \le \frac{2\gamma^3 t^2}{\epsilon^2 f\cdot\min\{\nu,s\}} \overset{\text{def}}{=} v.$$

After applying Theorem 21, it holds that

$$\mathbf{Pr}\Big[G \wedge \sum_{i\in[k]} D_i \le (1-\epsilon)ph(S)\Big] \le \exp\Big\{-\frac{t^2}{2v(1+\epsilon/3)}\Big\}$$
$$\le \exp\Big\{-\frac{\epsilon^2 f}{4\gamma^3(1+\epsilon/3)}\cdot\min\{k,s\}\Big\} \stackrel{\text{def}}{=} \epsilon_D(k).$$

The error probability on violating the upper bound on $D(S)$ can be computed in the same way, and yields the same as $\epsilon_D(k)$. To compute $\epsilon_{\mathcal{H}}$, we consider the union bound that $Q(S)$ and $D(S)$ yields good concentration. Specifically, let $G_{Q,D}(S)$ denote the event $(1-\epsilon)[1-(1+\delta)\gamma^2 f]^{\Delta}ph(S) < Q^i(S) \wedge (1-\epsilon)ph(S) < D^i(S) < (1+\epsilon)ph(S)$, it holds that

$$\mathbf{Pr}[\neg G_{Q,D}(S)] = 1-[1-\epsilon_Q(k)][1-\epsilon_D(k)]^2 \le \epsilon_Q(k)+2\epsilon_D(k)$$
$$\le \exp\Big\{\ln(\Delta+2) - \frac{\epsilon^2 f[1-(1+\delta)\gamma^2 f]^{\Delta+1}}{4\gamma^3\Delta(1+\epsilon/3)}\cdot\min\{k,s\}\Big\},$$

which defines $\epsilon_{\mathcal{H}}(k)$. □

Lemma 6. *For any chain $\mathcal{C}_i$ and any set J of consecutive adversarial queries and $\alpha(J,k)$ as defined in Equation (18), $\mathsf{goodConv}^i_{\mathcal{A}}(J,k)$ holds except with probability no more than $\epsilon_{\mathcal{A}}(k)$, where*

$$\epsilon_{\mathcal{A}}(k) \triangleq \exp\Big\{-\frac{\epsilon^2 f[1-(1+\delta)\gamma^2 f]^{\Delta}}{4\gamma^3\tau(1+5\epsilon)}\cdot k\Big\}.$$

Proof. For each $j \in J$, let A_j be equal to the difficulty of the block obtained with the j-th query as long as the target was at least $T(J)/\tau$; thus, $A(J) = \sum_{j\in J} A_j$. If $|J| = \nu$, identify J with $[\nu]$ and define the martingale

$$X_0 = 0;\ X_u = \sum_{j\in[u]} A_j - \sum_{j\in[u]} \mathbb{E}[A_i|\mathcal{E}_{j-1}], u \in [\nu].$$

For all $u \in [\nu]$ we have $X_u - X_{u-1} \le \tau/T(J)$, $\mathbf{Var}[X_u - X_{u-1}|\mathcal{E}_{u-1}] \le p\tau/T(J)$, and $E[A_j|\mathcal{E}_{j-1}] \le p$.

We now apply Theorem 21 with $b = \tau/T(J)$, $v = bp\nu \le bt/\epsilon$; and since $\epsilon < 1/2$ due to Condition (C1), we have $(1+4\epsilon)(1+\frac{\epsilon}{3}) < (1+5\epsilon)$ hence set

$$t = \max\Big\{\epsilon p\nu, 2(\frac{1}{\epsilon}+\frac{1}{3})b\cdot\frac{\epsilon^2 f[1-(1+\delta)\gamma^2 f]^{\Delta}}{4\gamma^3\tau(1+5\epsilon)}\cdot k\Big\}.$$

We have

$$\mathbf{Pr}\Big[\sum_{j\in J} A_j \ge p\nu + t\Big] \le \exp\Big\{-\frac{t}{2b(\frac{1}{3}+\frac{1}{\epsilon})}\Big\} \le \exp\Big\{-\frac{\epsilon^2 f[1-(1+\delta)\gamma^2 f]^{\Delta}}{4\gamma^3\tau(1+5\epsilon)}\cdot k\Big\},$$

which defines $\epsilon_{\mathcal{A}}(k)$. □

Given the two error probabilities above, we next show that the union probability that all ideal convergences hold in a sequence of k consecutive rounds is lower-bounded by $1-\epsilon_{\mathsf{conv}}(k)$.

Lemma 7. *In a (γ,s)-respecting environment, for any chain $\mathcal{C}_i$ and any set U of at least $k \ge \ell$ consecutive rounds in a* good *epoch, $\mathsf{goodConv}^i(U,\ell)$ holds except with probability no more than $\epsilon_{\mathsf{conv}}(k)$, where*

$$\epsilon_{\mathsf{conv}}(k) \triangleq k^2\cdot\max\{\epsilon_{\mathcal{A}}(\ell), \epsilon_{\mathcal{H}}(\ell)\}.$$

Proof. Fix a consecutive rounds $U' \subseteq U$ of at least ℓ rounds, and let J denote the adversarial queries in U'. It holds that $\mathbf{Pr}[\neg(\mathsf{goodConv}_{\mathcal{H}}(U') \wedge \mathsf{goodConv}_{\mathcal{A}}(J,\ell))] \leq 1-(1-\epsilon_{\mathcal{A}}(\ell))(1-\epsilon_{\mathcal{H}}(\ell)) \leq \epsilon_{\mathcal{A}}(\ell) + \epsilon_{\mathcal{H}}(\ell)$.

Therefore, consider the union probability for all subsequences in U, it holds that

$$\begin{aligned}\mathbf{Pr}[\neg\mathsf{goodConv}(U,\ell)] &= 1 - \prod_{|U'|\geq k}(1 - \mathbf{Pr}[\neg(\mathsf{goodConv}_{\mathcal{H}}(U') \wedge \mathsf{goodConv}_{\mathcal{A}}(J,\ell))]) \\ &\leq \sum_{|U'|\geq k} \mathbf{Pr}[\neg(\mathsf{goodConv}_{\mathcal{H}}(U') \wedge \mathsf{goodConv}_{\mathcal{A}}(J,\ell))] \\ &\leq \frac{(k-\ell+1)^2}{2}[\epsilon_{\mathcal{A}}(\ell) + \epsilon_{\mathcal{H}}(\ell)] \leq k^2 \max\{\epsilon_{\mathcal{A}}(\ell), \epsilon_{\mathcal{H}}(\ell)\},\end{aligned}$$

which defines $\epsilon_{\mathsf{conv}}(U,\ell)$. □

Thus, for a concrete value of the parameter ℓ, we set a ℓ such that $1 - \epsilon_{\mathsf{conv}}(\mathsf{M}) \leq 2\epsilon$. Recall that $\mathsf{M} = \Theta(\ell)$ (Equation (17)) thus there always exists $\ell = \mathcal{O}(1)$ that can satisfy this condition.

$$1 - \epsilon_{\mathsf{conv}}(\mathsf{M}) \leq 2\epsilon \tag{C3}$$

Theorem 8. *In a (γ, s)-respecting environment, if epoch ep is good, then there exist a subset $S \subseteq [m]$ of size $S > (1-\mu)m$ such that for all $i \in S$, $\mathsf{goodConv}^i(U,\ell)$ holds, where $U = \{(\mathsf{ep}-1)\cdot \mathsf{M}+1, \ldots, \mathsf{ep}\cdot\mathsf{M}\}$.*

Proof. Since the block target with which an honest party queries the RO depends on her local view of chains in epoch $\mathsf{ep}-1$ at the onset of epoch ep, the target of all honest queries in epoch ep is conditioned only on the execution in the previous epoch. Hence, all mining procedures in epoch ep are mutually independent.

We prove this theorem using Chernoff bound (Theorem 20). Let X_i denote the random variable such that if $\mathsf{goodConv}^i(U,\ell)$ holds then $X_i = 1$, and otherwise $X_i = 0$. Also, define $X = \sum_{i\in[m]} X_i$. Due to Lemma 7, $\mathbf{Pr}[X_i = 1] \geq 1 - \epsilon_{\mathsf{conv}}(k)$. We have $\mathbb{E}[X] \geq (1-\epsilon_{\mathsf{conv}}(k))m$. Hence, it holds that

$$\begin{aligned}\mathbf{Pr}[X \leq (1-\mu)m] &\leq \mathbf{Pr}\big[X \leq (1-\epsilon)(1-\epsilon_{\mathsf{conv}}(k))m\big] \\ &\leq \mathbf{Pr}\big[X \leq (1-\epsilon)\mathbb{E}[X]\big] \leq \exp\big\{-\epsilon^2(1-\epsilon_{\mathsf{conv}}(k))m/2\big\}.\end{aligned}$$

Since $m = \Theta(\log^2\kappa)$, the event that good convergence does not hold on sufficiently many chains occurs with probability negligible in κ the security parameter. □

Useful inequalities. We establish a few useful inequalities in the m-ary parallel chain setting. Note that, we have to apply these results carefully, only on executions where the m-ary parallel blockchains are mutually independent.

Lemma 9. *Let $U = \{u, \ldots, v\}$ be a set of consecutive rounds with length $|U| \geq \ell$, and let J be the set of adversarial queries in $U' = \{u - \Delta, \ldots, v + \Delta\}$.*

(a) $\mathsf{goodConv}^i_{\mathcal{H}}(U') \implies (1+\epsilon)p|J| \leq Q^i(U) \leq D^i(U)$.

(b) $\mathsf{goodConv}^i_{\mathcal{A}}(J,\ell) \implies A^i(J) < (1+\epsilon)p|J|$ *or* $T(J)\cdot A^i(J) < \frac{\epsilon \mathsf{M}_{\mathsf{IB}} f}{4\gamma^4}$, *and* $B^i(J) < (1+\epsilon)p|J|$ *or* $\tau T(J)\cdot B^i(J) < \frac{\epsilon \mathsf{M}_{\mathsf{IB}} f}{4\gamma^4}$.

Proof. For part (a), the second inequality follows directly from the definition. Regarding the first inequality, note that due to Lemma 2(b),

$$\begin{aligned}(1+\epsilon)p|J| &\le (1+\epsilon)(1-\delta)ph(U') \\ &\le (1+\epsilon)(1-\delta)(1+\frac{\epsilon^2}{2})ph(U) \le (1-\epsilon)[1-(1+\delta)\gamma^2 f]^{\Delta}ph(U).\end{aligned}$$

The second inequality is because $h(U') = h(U) + h(U' \setminus U)$ where $h(U' \setminus U) \le (2\gamma\Delta/\ell)h(U)$ and Condition (C1) applies. The last inequality is due to Condition (C2) and Definition 5.

For part (b), if $\tau\alpha(J) \le \epsilon p|J|$ then we are done. Otherwise, we have

$$T(J)\cdot A^i(J) < \left(1+\frac{1}{\epsilon}\right)\cdot\frac{\epsilon[1-(1+\delta)\gamma^2 f]^{\Delta} f\ell}{2(1+4\epsilon)\gamma^3} \le \frac{(1-\epsilon)^2 f}{2\gamma^3}\cdot\ell < \frac{\epsilon \mathsf{M}_{\mathsf{IB}} f}{4\gamma^4}.$$

The first inequality is by $p|J| < \tau\alpha(J)/\epsilon$ and then substituting $\alpha(J)$; the second one is due to the fact that $(1+4\epsilon)(1-\epsilon)^2 > 1+\epsilon$; and the last one is by Equation (17). For $B^i(J)$ it can be proved anaglously. □

5.3 'Good' Properties Hold

We first prove that if all the 'good' properties (Section 5.1) hold for the current epoch, then after applying the ApproxAgree function, the next epoch will be a concentrated epoch.

Lemma 10. GoodEpoch(ep) ∧ NoStaleChains(ep) ⟹ Concentrated-Epochs(ep + 1).

Proof. Consider the following three lemmas that addresses Equations (13) to (15) respectively.

Equation (13). Let $r = (\mathsf{ep}-1)\cdot\mathsf{M}+1$ and $U = \{r,\dots,\mathsf{ep}\cdot\mathsf{M}\}$; and let $w = \mathsf{ep}\cdot\mathsf{M}+1$ and $U' = \{w,\dots,(\mathsf{ep}+1)\cdot\mathsf{M}\}$. Let $\Delta_{\mathsf{ep}} = \max\{\mathsf{Diff}(\mathcal{C}), \mathcal{C}\in\mathcal{S}_r^i\in\mathcal{S}_r\} - \min\{\mathsf{Diff}(\mathcal{C}), \mathcal{C}\in\mathcal{S}_r^i\in\mathcal{S}_r\}$ denote the difference between the maximum and minimum chain difficulty in $\mathcal{S}_r$; and define $\Delta_{\mathsf{ep}+1}$ in the similar way with respect to $\mathcal{S}_w$. Let J denote the adversarial queries associated with U.

By Theorem 8, we learn that there exist a set $S\subseteq[m]$ of size $|S| > (1-\mu)\cdot m$ such that good properties as defined in Definition 5 holds. For any chain in $\mathsf{ep}+1$, by our target recalculation function and the definition of S, its difficulty is at least $\min\{\mathsf{Diff}(\mathcal{C})|\mathcal{C}\in\mathcal{S}_r^i\in\mathcal{S}_r\} + \min_{i\in S} Q^i(U)$ where $\min_{i\in S} Q^i(U) \ge (1-\epsilon)(1-(1+\delta)\gamma^2 f)^{\Delta}ph(U) \ge (1-\epsilon)^2 ph(U)$ (by Condition (C2)).

Regarding the upper bound, we let the adversary join force with the honest parties. Any chain at the onset of epoch $\mathsf{ep}+1$ is of difficulty upper bounded by $\max\{\mathsf{Diff}(\mathcal{C})|\mathcal{C}\in\mathcal{S}_r^i\in\mathcal{S}_r\} + \max_{i\in S}[Q^i(U)+A^i(J)]$ where $\max_{i\in S}[Q^i(U)+A^i(J)] \le (2-\delta)(1+\epsilon)ph(U)$.

By definition of Δ_{ep} and $\Delta_{\mathsf{ep}+1}$, we have

$$\begin{aligned}\Delta_{\mathsf{ep}+1} &\le \frac{\mu}{1-\mu}\cdot\Big(\Delta_{\mathsf{ep}} + [(2-\delta)(1+\epsilon) - (1-\epsilon)^2]ph(U)\Big) \\ &\le \frac{\mu}{1-\mu}\cdot\big(\Delta_{\mathsf{ep}} + ph(U)\big) \le \frac{\mu}{1-\mu}\cdot(3\mu+1)\gamma h(U') \le 3\mu ph(U').\end{aligned}$$

The first inequality follows Lemma 1(a), and the second one is due to Condition (C1). For the next inequality, we apply Lemma 2(a), and the last one is again derived from Condition (C1).

Equation (14). Let $r = (\mathsf{ep}-1)\cdot\mathsf{M}+1$ be the first round in epoch ep, and $w = \mathsf{ep}\cdot\mathsf{M}+1$ be the first round in epoch ep+1. Since ConcentratedEpochs(ep) holds, we learn that $T_r^{\min}\cdot(1+2\mu) \ge T_r^{\max}$. In addition, GoodChains(ep) implies that there is a set $S\subseteq[m]$ of size $|S| \ge (1-\mu)\cdot m$ with honest input-block majority and the ratio bounded as Equation (12); and NoStaleChains indicates that

all honest parties agree on these sets of input-blocks. For an alert party P in epoch $\mathsf{ep}+1$, let $T^{\mathsf{ref}}_{\mathsf{P}}$ and $\rho^{\mathsf{ref}}_{\mathsf{P}}$ denote the target and rate that P refers to when calculating her mining target. Consider our target recalculation function (Equation (9)), it holds that

$$\frac{T^{\max}_w}{T^{\min}_w} < \frac{\max_{\mathsf{P}} T^{\mathsf{ref}}_{\mathsf{P}}}{\min_{\mathsf{P}} T^{\mathsf{ref}}_{\mathsf{P}}} \cdot (1+\mu) \leq \Big(1 + \frac{\mu}{1-\mu} \cdot 2\mu\Big)(1+\mu) < 1 + 2\mu.$$

The first inequality is derived from Equation (12) and Equation (15); and the next one is by Lemma 1(b). The last inequality follows Condition (C1).

Equation (15). Note that in the first epoch parties mine with initial target T_0, we only consider epochs for $\mathsf{ep} \geq 2$. Let S denote the chain indices of size $|S| \geq (1-\mu)m$ that good property as Theorem 8 holds on all $i \in S$. Consider an alert party P in epoch $\mathsf{ep}+1$. Let $\rho_i = \mathsf{med}_{\mathtt{IB} \in \mathcal{C}_i} \mathsf{Diff}(\mathtt{IB}) / \sum_{\mathtt{IB} \in \mathcal{C}_i} \mathsf{Diff}(\mathtt{IB})$ denote the ratio computed by P on the i-th chain for $i \in S$. We identify two indices $i_{\min}$ and $i_{\max}$ such that $\rho_{i_{\min}}$ ($\rho_{i_{\max}}$ resp.) is the minimum (maximum resp.) among $\{\rho_i\}_{i \in S}$.

During epoch ep, honest parties produced input-blocks during rounds $U = \{u, \ldots, v\}$ where $u = (\mathsf{ep}-1) \cdot \mathsf{M} + \mathsf{M}_{\mathsf{FR}} + 1$ and $v = \mathsf{ep} \cdot \mathsf{M} - \mathsf{M}_{\mathsf{CP}}$. For chain index $i_{\min}$ ($i_{\max}$ resp.), let $\mathcal{B}$ ($\mathcal{B}'$ resp.) be the last block produced by honest parties before time v and denote its production time by w (w' resp.). Due to NoStaleChains, we get that $w > v - (\ell + 2\Delta)$ and $w' > v - (\ell + 2\Delta)$.

Consider the time interval that honest parties can contribute to the input-block set. For chain index $i_{\min}$, we denote it by $U_1 = \{u, \ldots, w - \Delta\}$; the earliest time step of U_1 is derived from the definition of u and the largest time is because it will take up to Δ rounds for all input-blocks to be diffused to and accepted by all alert parties. Next, for the time interval that the adversary can contribute on chain index $i_{\max}$, we denote it by $U_2 = [u - (2\ell + 4\Delta), w + \ell + 2\Delta]$. The bounds of U_2 are acquired by applying NoStaleChains and considering the unpredictability of an honest block. Let J denote all adversarial queries associated with U_2.

Let r denote the first round in epoch ep, we have:

$$\begin{aligned}\frac{\rho_{i_{\max}}}{\rho_{i_{\min}}} &\leq \frac{[D^{i_{\max}}(U) + A^{i_{\max}}(J)]/(1/T^{\max}_r)}{D^{i_{\min}}(U_1)/(1/T^{\min}_r)} \\ &< \frac{(1+\epsilon)p|J| + (1+\epsilon)ph(U)}{(1-\epsilon)ph(U_1)} \cdot \frac{T^{\max}_r}{T^{\min}_r} \leq \frac{(1+\epsilon)^2(2-\delta)}{(1-\epsilon)^2} \cdot (1+2\mu) \leq 2 - \mu.\end{aligned}$$

Thus, given this bound, it holds that

$$\frac{\max_{\mathsf{P} \in \mathcal{H}^{\mathsf{ep}}} \rho^{\mathsf{ref}}_{\mathsf{P}}}{\min_{\mathsf{P} \in \mathcal{H}^{\mathsf{ep}}} \rho^{\mathsf{ref}}_{\mathsf{P}}} \leq 1 + \frac{\mu}{1-\mu} \cdot \Big(\frac{\rho_{i_{\max}}}{\rho_{i_{\min}}} - 1\Big) \leq 1 + \mu.$$

The first inequalities is due to NoStaleChains and Lemma 1(b), and the second one follows that Condition (C1). □

Then, we show that not too many parallel chains can be 'stale' given a steady block generation rate in the current and all previous epochs.

Lemma 11. *Assume* GoodEpoch(ep) *and* $\mathsf{goodConv}^i(U, \ell)$ *hold for* $U = \{(\mathsf{ep}-1) \cdot \mathsf{M} + 1, \ldots, \mathsf{ep} \cdot \mathsf{M}\}$. *Then for any* $r \in U$ *and any* $\mathcal{C} \in \mathcal{S}^i_r$, *the first block* $\mathcal{B}$ *on* $\mathcal{C}$ *in epoch* ep *is mined with target* T *such that* $T^{\max}/(1+2\mu) \leq T \leq T^{\min} \cdot (1+2\mu)$.

Proof. Suppose—towards a contradiction, there is a chain $\mathcal{C}^* \in \mathcal{S}^i_r$ for a round $r \in U$ such that the first block $\mathcal{B}^*$ on $\mathcal{C}^*$ yields difficulty either $T < T^{\max}/(1+2\mu)$ or $T > T^{\min} \cdot (1+2\mu)$. Let r be the first round that such $\mathcal{C}^*$ exists.

First, we argue that $\mathcal{C}^*$ has the following property. Let $(\overline{\mathcal{C}}_i)_{i\in[m]} = \mathsf{RefChains}(\mathcal{B}^*)$ denote the set of chains where $\mathcal{B}^*$ is the first block in epoch ep on $\mathcal{C}$. Let S^{ep} be the subset of chains with good properties as specified in Theorem 8 at the onset of ep. Then, for at least one $i \in S^{\mathsf{ep}}$, $\overline{\mathcal{C}}_i$ forks from the chain referred by honest parties for at least $\mathsf{M_{CP}}$ rounds; otherwise, we apply Equation (14) in Lemma 10 which contradicts our assumption.

We show that $\mathcal{C}^*$ can never compete with any other chain in $\mathcal{S}^i_r$ for any $r \in U$. Towards this goal, we define a round u that the adversary start to create forks contributing to the difficulty of $\mathcal{C}^*$. Specifically, let u be the timestamp of an honest block $\mathcal{B}_u$ in epoch ep' such that, there is a set of chains $\mathcal{C}^{\mathsf{ep}'}, \ldots, \mathcal{C}^*$ such that (i) there is no honest block with timestamp larger than u; and (ii) $\mathcal{C}^{\mathsf{ep}'} \in \mathsf{RefChains}(\mathcal{C}^{\mathsf{ep}'+1})$ and $\mathcal{C}^{\mathsf{ep}'}$ is neither the μm maximum nor μm minimum chains in $\mathsf{RefChains}(\mathcal{C}^{\mathsf{ep}'+1})$. Thus, we learn that such list of chains contribute to the accumulated difficulty of $\mathcal{C}^*$, but has not been extended by honest parties since $\mathcal{B}_u$.

Let $V = \{u, \ldots, r\}$ and J the associated adversarial queries with V. Suppose V spans n epochs from $\mathsf{ep}-n+1$ to ep, we partition V into n sections $V_1, \ldots, V_n$ such that (i) for the first partition, if u is the last $\ell + 2\Delta$ rounds then $V_1 = \emptyset$, otherwise $V_1 = \{u, \ldots, (\mathsf{ep}-n+1)\cdot\mathsf{M} - \Delta\}$; (ii) for the last partition, if r is the first $\ell + 2\Delta$ rounds then $V_n = \emptyset$, otherwise $V_n = \{(\mathsf{ep}-1)\cdot\mathsf{M} + \Delta, \ldots, r\}$; and (iii) for the rest, $V_i = \{(\mathsf{ep}-n+i)\cdot\mathsf{M} + \Delta, \ldots, (\mathsf{ep}-n+i+1)\cdot\mathsf{M} - \Delta\}$. By our chain difficulty rule and applying Lemma 4 in each epoch, we learn that for any chain $\mathcal{C} \in \mathcal{S}^i_r$, $\mathsf{Diff}(\mathcal{C}) \geq \mathsf{Diff}(\mathcal{C}^{(u)}) + (1-\epsilon)[1-(1+\delta)\gamma^2 f]^{\Delta} p \sum_{i\in[n]} h(V_i)$, where $\mathcal{C}^{(u)}$ is the chain ends with $\mathcal{B}_u$ as identified above.

Let $d = \mathsf{Diff}(\mathcal{C}^*) - \mathsf{Diff}(\mathcal{C}^{(u)})$. It suffices to show that $d < (1-\epsilon)[1-(1+\delta)\gamma^2 f]^{\Delta} p \sum_{i\in[n]} h(V_i)$. We partition J into n sections $(J_1, \ldots, J_n)$, where the i-th section corresponds to the queries with epoch index $\mathsf{ep}-n+i$. Since $\mathsf{goodConv}$ holds for all epochs on at least $(1-\mu)m$ chains, combining our argument for Equation (13) in Lemma 10, we have $d \leq \sum_{i\in[n]} A^*(J_i)$, where $A^*(J_i)$ is associated with the index of the i-th chain when identifying $\mathcal{B}_u$.

Since we filter the μm chains of maximum and minimum difficulty in that epoch, for each J_i we learn that either $A^*(J) < (1+\epsilon)p|J|$ or the number of blocks is bounded as Lemma 9. For the latter case, we show that $A^*(J_i)$ is bounded by respect to another $A^*(J_{i'})$—on $\mathcal{C}^{\mathsf{ep}'+i'-1}$, where the number of input-blocks produced within $A^*(J_{i'})$ is at least $\mathsf{M_{IB}} f/\tau$. Such $A^*(J_{i'})$ must exist, as otherwise the mining difficulty keeps decreasing exponentially and by applying Lemma 22 we learn that $d < (1-\epsilon)[1-(1+\delta)\gamma^2 f]^{\Delta} p[h(V_1) + h(V_2)]$.

For a sequence of partitions $J_{u.+1}, \ldots, J_v$, we associate them with J_u which has at least $\mathsf{M_{IB}} f/\tau$ blocks. By Equation (17) and Lemma 22, we have $\sum_{i=u+1}^{j} A^*(J_i) < \epsilon/(1-1/\tau) A^*(J_u) < 2\epsilon A^*(J_u)$. Thus, it holds that

$$d \leq \sum_{i\in[n]} (1+2\epsilon)(1+\epsilon)p|J_i| < (1-\delta+4\epsilon)ph(V) < (1-\epsilon)[1-(1+\delta)\gamma^2 f]^{\Delta} p \sum h(V_i).$$

The first inequality holds due to our discussions on J_i; the second one is by our honest majority assumption; and the last one is due to Lemma 2(b) and Condition (C1). □

Lemma 12. *Assume (i)* GoodEpoch(ep) *holds; (ii) there is an honestly generated block $\mathcal{B}$ with timestamp $r \in \{(\mathsf{ep}-1)\cdot\mathsf{M}, \ldots, (\mathsf{ep}-1)\cdot\mathsf{M} + \mathsf{M_{FR}}\}$ in any $\mathcal{C} \in \mathcal{S}^i_w$ for $r \leq w < \mathsf{ep}\cdot\mathsf{M}$; and (iii) $\mathsf{goodConv}^i(U, \ell)$ holds for $U = \{r, \ldots, \mathsf{ep}\cdot\mathsf{M}\}$. Then* NoStaleChains($i, \mathsf{ep}$) *holds.*

Proof. First, note that at any round up to $r + (\ell + 2\Delta)$, $\mathcal{C}$ is not stale due to block $\mathcal{B}$. For the rest of the rounds, assume for the sake of contradiction that w is the first round that $\mathcal{C}$ is stale, and

let $U = \{r+1, \ldots, w\}$ and $U = \{r+\Delta, \ldots, w-\Delta\}$, and J the adversarial queries in U (we have $|U| > \ell$). We are going to show the contradiction by proving that $Q(U') > A(J)$.

If $A(J) < (1+\epsilon)p|J|$, then we obtain $A(J) < Q(U')$ from Lemma 9(a). Otherwise, $A(J) < (1+\frac{1}{\epsilon})\tau\alpha(J,\ell)$ and we associate J with $T(J) = T_r^{\min}$, where $\mathcal{B}$ is the first block on $\mathcal{C}$ in epoch ep. However, by considering the first ℓ rounds in U it holds that

$$\begin{aligned} Q(U) &> (1-\epsilon)[1-(1+\delta)\gamma^2 f]^{\Delta} \cdot \frac{ph_r \ell T_r^{\min}}{\gamma T_r^{\min}} \\ &\geq \frac{(1-\epsilon)[1-(1+\delta)\gamma^2 f]^{\Delta} f\ell}{2\gamma^2 T(J)} > (1+\frac{1}{\epsilon})\tau\alpha(J,\ell) > A(J). \end{aligned}$$

The first inequality is by the definition of $Q(U)$, and the third on is by definition of $\alpha(J,\ell)$ in Equation (18). □

Lemma 13. *Assume* GoodEpoch *holds for epoch* ep. *Then there is a subset* $S \subseteq [m]$ *of size* $|S| > (1-\mu)m$ *such that for any* $i \in S$ *and any* $r \in \{(\mathsf{ep}-1)\cdot\mathsf{M}+\mathsf{M}_{\mathsf{FR}}+1, \ldots, \mathsf{ep}\cdot\mathsf{M}\}$, *and for any* $\mathcal{C} \in \mathcal{S}_r^i$ *there is at least one honestly generated block with timestamp* $w \in \{(\mathsf{ep}-1)\cdot\mathsf{M}+\mathsf{M}_{\mathsf{FR}} - (\ell+2\Delta), \ldots, (\mathsf{ep}-1)\cdot\mathsf{M}+\mathsf{M}_{\mathsf{FR}}\}$.

Proof. Suppose—towards a contradiction—that there is a subset of chains with size no less than μm which do not contain any honestly generated block with timestamp in $\{(\mathsf{ep}-1)\cdot\mathsf{M}+\mathsf{M}_{\mathsf{FR}} - (\ell+2\Delta), \ldots, (\mathsf{ep}-1)\cdot\mathsf{M}+\mathsf{M}_{\mathsf{FR}}\}$. We are going to to show that we can identify a set $S \subseteq [m]$ of chain indices with size $|S| > (1-\mu)m$ that do contain such block.

Let S denote the set of chain indices that good convergence probability holds as Theorem 8 and fix a chain index $i \in S$. It suffices to prove that such block $\mathcal{B}$ do exist on chain $\mathcal{C}$. We prove this by contradiction: Suppose there is a chain $\mathcal{C}'$ that does not include $\mathcal{B}$ such that $\mathsf{Diff}(\mathcal{C}') > \mathsf{Diff}(\mathcal{C})$ at some rounds $w \geq (\mathsf{ep}-1)\cdot\mathsf{M}+\mathsf{M}_{\mathsf{FR}}$. We consider three cases.

First, there is another honest block on $\mathcal{C}'$ with timestamp $r' < (\mathsf{ep}-1)\cdot\mathsf{M}+\mathsf{M}_{\mathsf{FR}} - (\ell+2\Delta)$. Then, Lemma 12 shows that there is at least one honest block for every $\ell+2\Delta$ rounds after r'. Thus at least one honest block in $\{(\mathsf{ep}-1)\cdot\mathsf{M}+\mathsf{M}_{\mathsf{FR}} - (\ell+2\Delta), \ldots, (\mathsf{ep}-1)\cdot\mathsf{M}+\mathsf{M}_{\mathsf{FR}}\}$.

Second, there is no honest block on $\mathcal{C}'$ with timestamp in epoch ep, yet $\mathsf{Diff}(\mathcal{C}') - \sum_{\mathcal{B}\in\mathcal{C}'^{\mathsf{ep}}} \mathsf{Diff}(\mathcal{B}) \leq \max\{\mathsf{Diff}(\mathcal{C}'') | \mathcal{C}'' \in \mathcal{S}^i_{(\mathsf{ep}-1)\cdot m}\}$. Let $U = \{(\mathsf{ep}-1)\cdot\mathsf{M}+1, \ldots, r\}$ and J the associated adversarial queries with U; let $U' = \{(\mathsf{ep}-1)\cdot\mathsf{M}+1, \ldots, \mathsf{ep}\cdot\mathsf{M}\}$. Recall Equation (13) in Lemma 10, we have $A^i(J) + \Delta_{\mathsf{ep}} \geq Q^i(U)$ for all $i \in [m] \setminus S$. However, due to Lemma 7, for at least one chain index $i^* \in [m] \setminus S$, we have the following contradiction.

$$Q^i(U) < A^i(J) + \Delta_{\mathsf{ep}} < (1+\epsilon)p|J| + 3\mu ph(U') \leq [(1+\epsilon)(1-\delta) + 2\mu]ph(U) \leq Q^i(U).$$

Third, there is no honest block on $\mathcal{C}'$ with timestamp in epoch ep, and $\mathsf{Diff}(\mathcal{C}') - \sum_{\mathcal{B}\in\mathcal{C}'^{\mathsf{ep}}} \mathsf{Diff}(\mathcal{B})$ is of more difficulty than any chain $\mathcal{C}'' \in \mathcal{S}^i_{(\mathsf{ep}-1)\cdot m}$. We perform the same partition as Lemma 11 and conclude that no adversarial fork can compete with chains in $\mathcal{S}^i_w$. □

Lemma 14. GoodEpoch(ep) $\implies$ NoStaleChains(ep).

Proof. Suppose—towards a contradiction—that there is a subset of chains with size larger than μm which chains are stale at some rounds in $\{(\mathsf{ep}-1)\cdot\mathsf{M}+\mathsf{M}_{\mathsf{FR}}+1, \ldots, \mathsf{ep}\cdot\mathsf{M}\}$. We are going to to show that we can identify a set $S \subseteq [m]$ of chain indices with size $|S| > (1-\mu)m$ that are not stale in any of these rounds.

Let S denote the set of chain indices that good convergence probability holds as Theorem 8. For any chain index $i \in S$, from Lemma 13 we learn that, for any chain $\mathcal{C}$ that honest parties

try to extend, there is at least one honestly generated block $\mathcal{B}$ on $\mathcal{C}$ with timestamp $r \in \{(\mathsf{ep}-1)\cdot\mathsf{M}+\mathsf{M}_{\mathsf{FR}}-(\ell+2\Delta),\ldots,(\mathsf{ep}-1)\cdot\mathsf{M}+\mathsf{M}_{\mathsf{FR}}\}$. Then, we apply Lemma 12 and learn that NoStaleChains(i,ep) holds. Thus, no chain in S is stale for any round in the last $\mathsf{M}_{\mathsf{IB}}+\mathsf{M}_{\mathsf{CP}}$ rounds in ep, which contradicts our assumption and concludes the proof. □

Next, we show that if, in an epoch, the adversary can only manage to make a small fraction of parallel chains stale, then the set of input-blocks included on the rest of chains will have good properties. Namely, their median target will be well-bounded by honest mining targets, and the ratio between the expected difficulty versus the actual accumulated difficulty will be bounded as in Equation (12). Given that honest parties share a common view on these input-blocks, we argue that in the next round, their mining target will be set appropriately, thus yielding steady block generation rates as defined in Equations (10) and (11).

Lemma 15. GoodEpoch(ep) $\wedge$ NoStaleChains(ep) $\implies$ GoodChains(ep).

Proof. Let S denote the chain indices of size $|S| \geq (1-\mu)m$ that good property as Theorem 8 holds on all $i \in S$. Due to NoStaleChains(ep) and Lemma 14, we learn that for any chain index $i \in S$, there are no stale chains in the last $\mathsf{M}-\mathsf{M}_{\mathsf{FR}}$ rounds. We are going to show that good chains holds for all $i \in S$.

Fix a chain $\mathcal{C} \in \mathcal{S}^i_{\mathsf{ep}}$, consider the input-block set B of all input-blocks included in $\mathcal{C}^{\mathsf{ep}}$. We prove that (i) honestly generated input-blocks weigh more than the rest; and (ii) the ratio between the total weight and median input-block difficulty is well-bounded.

During epoch ep, honest parties produced input-blocks during rounds $U = \{u,\ldots,v\}$ where $u = (\mathsf{ep}-1)\cdot\mathsf{M}+\mathsf{M}_{\mathsf{FR}}+1$ and $v = \mathsf{ep}\cdot\mathsf{M}-\mathsf{M}_{\mathsf{CP}}$. Let $\mathcal{B}$ be the last block produced by honest parties before time v and denote its production time by w. Due to Lemma 14, we get that $w > v-(\ell+2\Delta)$.

Consider the time interval that honest parties can contribute to the input-block set. We denote it by $U_1 = \{u,\ldots,w-\Delta\}$; the earliest time step of U_1 is derived from the definition of u and the largest time is because it will take up to Δ rounds for all input-blocks to be diffused to and accepted by all alert parties. Next, for the time interval that the adversary can contribute, we denote it by $U_2 = [u-(2\ell+4\Delta), w+\ell+2\Delta]$. The bounds of U_2 are acquired by applying Lemma 14 and considering the unpredictability of an honest block.

Honest majority. In order to prove that alert parties can produce at least half of input-blocks, it suffices to show that $D(U_1) > A(J)$ where J are the adversarial queries associated with U_2. Towards this goal, we first show that the number of queries alert parties can make during U_2 is at most 2ϵ more than those in U_1:

$$\begin{aligned} h(U_2) &\leq \left(1+\frac{\gamma|U_2\setminus U_1|}{U_1}\right)h(U_1) \\ &\leq \left(1+\frac{\gamma(3\ell+7\Delta)}{\mathsf{M}_{\mathsf{IB}}-(\ell+3\Delta)}\right) \leq \left(1+\frac{\epsilon}{1-\epsilon/2\gamma}\right)h(U_1) < (1+2\epsilon)h(U_1). \end{aligned}$$

The first inequality is due to Lemma 2(b); and the next one is by the definition of U_1, U_2. The third inequality is from Equation (17) ($\ell+2\Delta \leq \epsilon\mathsf{M}_{\mathsf{IB}}/4\gamma$); and the last one is by Condition (C1). Hence, we have

$$D^i(U_1) \geq (1-\epsilon)ph(U_1) > (1-3\epsilon)ph(U_2) > A^i(J).$$

The first inequality is from our choice of the chain index; and the second one is achieved by substituting $h(U_1)$ with $h(U_2)$. The last inequality follows the honest majority assumption and Condition (C1).

Lower bound. We show that alert parties can alone produce enough input-blocks w.r.t. the median weight. Let $T^{\mathsf{med}} = \mathsf{Diff}(\mathtt{IB})$ where $\mathtt{IB}$ yields the median difficulty in $\mathcal{C}$; also let h^* denote the number of honest parties in the first round in epoch ep.

$$D^i(U_1) > (1-\epsilon)ph(U_1) \geq (1-\epsilon)\frac{ph^*|U_1|}{\gamma} \geq (1-\epsilon)^2 \cdot \frac{\mathsf{M}_{\mathsf{IB}} \cdot f}{2\gamma^2 T^{\min}} \geq \frac{\mathsf{M}_{\mathsf{IB}} \cdot f}{2(1+\delta)\gamma^2} \cdot \frac{1}{T^{\mathsf{med}}}.$$

The first inequality holds because of our choice on chain index. The second one comes from Lemma 2(a). The third inequality follows the fact that ep is a good epoch (Equation (10)) and the lower bound on the duration of U. The last one is due to Condition (C1) and honest majority among input blocks (thus $T^{\min} \leq T^{\mathsf{med}}$).

Upper bound. We show that even if all parties join force they cannot produce too many input-blocks w.r.t. the median weight. For honest parties, since ep is good, recall Lemma 2(a) and Equation (10), $ph(U) \leq p\gamma h^*|U| \leq (1+\delta)\gamma^2 f|U|/T^{\max}$. Thus, we have

$$D^i(U) < (1+\epsilon)ph(U) \leq (1+\epsilon)(1+\delta)\gamma^2 f|U| \cdot \frac{1}{T^{\max}} \leq (1+\epsilon)(1+\delta)\gamma^2 f|U| \cdot \frac{1}{T^{\mathsf{med}}}.$$

Regarding the adversary, if $T(J)A^i(J) \leq \tau T^{\mathsf{med}} A^i(J) \leq \epsilon(1+\delta)\mathsf{M}_{\mathsf{IB}} f/(2\gamma^2)$, then the total number of blocks is less than $2(1+\delta)\gamma^2 \mathsf{M}_{\mathsf{IB}} f$. Otherwise,

$$\begin{aligned} A^i(J) &< (1+\epsilon)p|J| < (1+\epsilon)(1-\delta)ph(U_2) \\ &\leq (1+\epsilon)(1+\delta)(1-\delta)\gamma^2 f|U_2| \cdot \frac{1}{T^{\max}} \leq (1+\epsilon)^2(1+\delta)(1-\delta)\gamma^2 \mathsf{M}_{\mathsf{IB}} f \cdot \frac{1}{T^{\mathsf{med}}}. \end{aligned}$$

The second inequality holds due to the δ gap between the number of honest and adversarial queries per round; and the next one holds by applying Lemma 2(a) with the first round in this good epoch (also with Equation (10)).

Then, Condition (C1) implies that $(1+\epsilon)^2(2-\delta) < 2$—i.e., the ratio between the accumulated difficulty of all input blocks and $1/T^{\mathsf{med}}$ is upper-bounded by $2(1+\delta)\gamma^2 \mathsf{M}_{\mathsf{IB}} f$. □

Lemma 16. $\textsc{GoodEpoch}(\mathsf{ep}) \wedge \textsc{GoodChains}(\mathsf{ep}) \implies \textsc{GoodEpoch}(\mathsf{ep}+1)$.

Proof. We denote $r = (\mathsf{ep}-1)\cdot \mathsf{M}+1$ as the first round of epoch ep. Let $w = \mathsf{ep}\cdot \mathsf{M}+1$ be the first round of $\mathsf{ep}+1$, we first show that Equation (10) holds for w. Note that, since $\textsc{GoodChains}(\mathsf{ep})$ holds, there is a subset $S \subseteq [m]$ ($|S| > (1-\mu)m$) of chains with the following property: honest parties contribute more than half of the input blocks included; and for these input blocks, the ratio between their total difficulty and median difficulty is well-bounded. Let ρ_i denote the ratio between input-block total difficulty and median difficulty on chain index i in epoch ep, and let $\rho^{\max}$ ($\rho^{\min}$ resp.) denote the maximum (minimum resp.) among indices $i \in S$.

Consider an alert party P at round w and let $T_{\mathsf{P}}^{\mathsf{ref}}$ denote her reference block (recall Equation (7)). Since ep is good, it holds that, for any alert party P, $f/2\gamma \leq ph_r T_{\mathsf{P}}^{\mathsf{ref}} \leq (1+\delta)\gamma f$.

Lower bound in Equation (10). If $\rho^{\max} \leq \mathsf{M}_{\mathsf{IB}} f/\gamma$, due to Equation (9) we learn that for any alert party P, its target T_w set for epoch $\mathsf{ep}+1$ satisfies $T_w \geq \gamma T_{\mathsf{P}}^{\mathsf{ref}}$. Further, $\rho^{\max} \leq \mathsf{M}_{\mathsf{IB}} f/\gamma$ implies that there is a chain index $i^* \in S$ such that, for its median-difficulty input-block target T^* and any alert party P, $T_w \geq T_w^{\min} \geq \gamma T^*$. Hence, $ph_w T_w^{\min} \geq ph_r T_w^{\min}/\gamma \geq ph_r T^* \geq ph_r T_r^{\min} \geq f/2\gamma$, where the second last inequality is due to honest input-block majority on the i^*-th chain.

If not, assume that for a chain index $i^{\max} \in S$, $\rho^{\max} = \rho_{i^{\max}} > \mathsf{M}_{\mathsf{IB}} f/\gamma$. Consider the party P that queries $T_w^{\min}$ at round w, we write i^* the chain index of her reference target (possibly $i^* = i^{\max}$). Let u, v denote the first and last round in epoch ep such that alert parties contribute to the set of

input-blocks on the i^*-th chain. Consider $U = \{u, \ldots, v\}$ and $U' = \{u - 2(\ell + 2\Delta), \ldots v + \ell + 3\Delta\}$. Let J denote the set of queries available to the adversary in U'. From the proof of Lemma 15, we learn that all adversarial input-blocks are generated during queries in J. Further, by Equation (17), $|U| < |U'| \leq (1+\epsilon)\mathsf{M}_{\mathsf{IB}}$; and by Lemma 2(a), $h(U') \leq \gamma h_w |U'|$ and $h(U) \leq \gamma h_w |U|$. We have:

$$A^{i^*}(J) < (1-\delta)(1+\epsilon)ph(U') \leq (1-\delta)(1+\epsilon)^2 p\gamma h_w \mathsf{M}_{\mathsf{IB}},$$

and $D^{i^*}(U) < (1+\epsilon)ph(U) \leq (1+\epsilon)p\gamma h_w \mathsf{M}_{\mathsf{IB}}$. Since P queries $T_w^{\min}$ at round w, we learn that $\mathsf{M}_{\mathsf{IB}} = (T_w^{\min}/T_{\mathsf{P}}^{\mathsf{ref}})(1/f)\rho_{\mathsf{P}}^{\mathsf{ref}}$. Assume $ph_w T_w^{\min} < f/2\gamma$, we get the following contradiction.

$$\begin{aligned} 2p\gamma h_w \mathsf{M}_{\mathsf{IB}} &\leq 2p\gamma h_w \cdot \frac{T_w^{\min}}{T_{\mathsf{P}}^{\mathsf{ref}}} \cdot \frac{1}{f} \cdot \rho_{\mathsf{P}}^{\mathsf{ref}} < \frac{1}{T_{\mathsf{P}}^{\mathsf{ref}}} \cdot \rho_{\mathsf{P}}^{\mathsf{ref}} \leq \frac{1}{T_{\mathsf{P}}^{\mathsf{ref}}} \cdot \rho^{\max} \\ &\leq (1+\mu)[D^{i^*}(U) + A^{i^*}(J)] < (2-\delta)(1+\mu)(1+\epsilon)^2 p\gamma h_w \mathsf{M}_{\mathsf{IB}} < 2p\gamma h_w \mathsf{M}_{\mathsf{IB}}. \end{aligned}$$

The third inequality is derived from Equation (15), and the last one follows Condition (C1).

Upper bound in Equation (10). If $\rho^{\min} \geq \gamma \mathsf{M}_{\mathsf{IB}} f$, Equation (9) implies that for any alert party P, its target T_w set for epoch $\mathsf{ep}+1$ satisfies $T_w \leq T_{\mathsf{P}}^{\mathsf{ref}}/\gamma$. Note that, $\rho^{\min} \geq \gamma \mathsf{M}_{\mathsf{IB}} f$ implies that there is a chain index $i^* \in S$ such that, when considering its median-difficulty input-block target T^* and any alert party P, we have $T_w \leq T_w^{\max} \leq T^*/\gamma$. Hence, $ph_w T_w^{\max} \leq p\gamma h_r T_w^{\max} \leq ph_r T^* \leq ph_r T_r^{\max} \leq (1+\delta)\gamma f$, where the second last inequality is due to honest input-block majority on the i^*-th chain.

Otherwise, assume that for a chain index $i^{\min} \in S$, $\rho^{\min} = \rho_{i^{\min}} < \gamma \mathsf{M}_{\mathsf{IB}} f$. Consider the party P that queries $T_w^{\max}$ at round w, we learn that $\mathsf{M}_{\mathsf{IB}} = (T_w^{\max}/T_{\mathsf{P}}^{\mathsf{ref}})(1/f)\rho_{\mathsf{P}}^{\mathsf{ref}}$. Let $U = \{u, \ldots, v\}$ denote the interval in epoch ep such that alert parties contribute to the set of input-blocks on the $i^{\min}$-th chain; also we write $T^{i_{\min}}$ the input-block target of median difficulty on this chain. Assume $ph_w T_w^{\max} > (1+\delta)\gamma f$, we get the following contradiction.

$$\begin{aligned} \frac{ph_w \mathsf{M}_{\mathsf{IB}}}{(1+\delta)\gamma} &= \frac{ph_w}{(1+\delta)\gamma} \cdot \frac{T_w^{\max}}{T_{\mathsf{P}}^{\mathsf{ref}}} \cdot \frac{1}{f} \cdot \rho_{\mathsf{P}}^{\mathsf{ref}} > \frac{1}{T_{\mathsf{P}}^{\mathsf{ref}}} \cdot \rho_{\mathsf{P}}^{\mathsf{ref}} \geq \frac{1}{T_{\mathsf{P}}^{\mathsf{ref}}} \cdot \rho^{\min} \\ &\geq (1-2\mu)Q(U) \geq (1-2\mu)(1-\epsilon)(1-(1+\delta)\gamma^2 f)^{\Delta} \cdot \frac{ph_w|U|}{\gamma} \geq \frac{ph_w \mathsf{M}_{\mathsf{IB}}}{(1+\delta)\gamma}. \end{aligned}$$

The third inequality is derived from Equation (14) and that $T_{\mathsf{P}}^{\mathsf{ref}} \leq (1+2\mu)T^{i_{\min}}$, and the last one follows Condition (C1).

Bounds in Equation (11). For any round $u \in \{\mathsf{ep}\cdot\mathsf{M}+1, \ldots, (\mathsf{ep}+1)\cdot\mathsf{M}\}$, we have $h_w/\gamma \leq h_u \leq \gamma h_w$. Since $\mathsf{M} < s$, combining this inequality with $f/2\gamma \leq ph_w T_w^{\min} \leq ph_w T_w^{\max} \leq (1+\delta)\gamma f$ we obtain the desired inequality. □

Finally, we are able to prove our intended result, which follows directly from Lemmas 14 to 16.

Theorem 17. *In a (γ, s)-respecting environment, if* NoStaleChains, GoodEpochs, GoodChains, ConcentratedEpochs *hold for epoch* ep*, then they also hold for epoch* $\mathsf{ep}+1$.

5.4 Consensus and SMR with Fast Difficulty Adjustment

We are now ready to show how to build permissionless consensus and state machine replication (a.k.a., distributed ledgers) on top of our parallel chains. Compared with previous results [GKS24, GKS25], our protocol is able to adapt to fast party fluctuation.

Epoch oblivious agreement. It has been shown in Chain-King Consensus [GKS24] that, PoW-based consensus with fast termination can be built generically on top of *epoch oblivious agreement*, where parties reach agreement on the set of input-blocks on a sufficiently large fraction of chains, while the oblivious aspect indicates that they will not be able to figure out on which subset do they agree. Specifically, parties execute a state update algorithm at the end of each epoch (inspired from classical phase-king consensus [BGP89]), and epoch oblivious agreement guarantees that parties terminate with constant probability. With fast PoW-based consensus, [GKS24] presents how to build state machine replication with fast transaction settlement with generic usage of parallel blockchains.

Thus, it suffices for us to show that our parallel blockchain can implement epoch oblivious agreement. Since we have employed input-blocks in our protocol, we let these input-blocks include the same input messages as per Chain-King Consensus in the `val` field. In Lemma 18, we claim that, our protocol achieves oblivious agreement in each epoch. The proof of this lemma is straightforward: Agreement follows NoStaleChains (Lemma 14) and honest input-block majority follows GoodChains (Lemma 15).

Lemma 18 (Epoch-oblivious Agreement). *Let* P, P' *be two parties that are alert at the end of epoch* ep*, and let* $\mathbb{C}, \mathbb{C}'$ *denote the parallel chains held by* P, P' *respectively. There exists a subset* $S \subseteq [m]$ *of size* $|S| > (1-\mu) \cdot m$ *such that for all* $i \in S$*, then the following two properties hold on chains* $\mathcal{C} = \mathbb{C}_i^{\mathsf{ep}}$ *and* $\mathcal{C}' = \mathbb{C}_i'^{\mathsf{ep}}$*.*

- **Agreement:** $\mathcal{C}^{\mathsf{ep}\cdot\mathsf{M}-\mathsf{M}_{\mathsf{CP}}\rceil} = \mathcal{C}'^{\mathsf{ep}\cdot\mathsf{M}-\mathsf{M}_{\mathsf{CP}}\rceil}$.
- **Honest input-block majority:** *For all input-blocks included in* $\mathcal{C}$ *and* $\mathcal{C}'$*, more than half of them are produced by alert parties.*

View retrieval for fresh parties. Alert parties that are always online will sync with each other by periodically performing the state update operations at the end of each epoch. Yet, as the adversary can keep extending chains in previous epochs, a fresh new party might get diverged when joining the protocol. Hence we discuss how new parties can securely bootstrap and retrieve previous views.

Recall that we assume a global clock $\mathcal{G}_{\mathsf{Clock}}$, thus a fresh party P learns the exact time upon joining the protocol. In order to bootstrap, party P first passively listens to the protocol execution until the conclusion of an epoch. In the meantime, P keeps updating her local parallel blockchains using the same heaviest chain selection rule as alert parties. We learn that when concluding the epoch ep which P joins, P enjoys oblivious agreement with all alert parties over their views of epoch ep. Then, the next goal is for P to retrieve obliviously honest views in previous epochs. This can be achieved by letting honest parties reporting their local view of the previous epoch using input-blocks, and then the fresh joining party P retrieve the old views using weak agreement, starting from epoch $\mathsf{ep}-1$ to the first epoch. We refer to [GKS24, GKS25] for more details on the view retrieval mechanism on parallel blockchains.

After concluding the bootstrapping phase, which takes up to an epoch (i.e., a constant number of rounds), the fresh joining party P bootstraps her local parallel chain and syncs her internal states with other parties, thus turning P into an alert party.

Finally, we conclude that, when appropriately parameterized, there is a ledger protocol that satisfies Consistency and expected-constant Liveness, while tolerating fast party fluctuation (i.e., the party fluctuation window is of constant length).

Theorem 19. *In a* (γ, s)*-respecting environment, assume that honest majority holds throughout the execution (Definition 2), the diffusion network is* Δ*-bounded, and the fluctuation window is of*

length $s = \mathsf{M} + 2\Delta = \mathcal{O}(1)$ *where* M *is the duration of a protocol epoch. If all protocol parameters are well set according to Conditions* (C1), (C2) *and* (C3), *then:*

- *There is a Consensus (Definition 3) protocol that achieves Agreement and Validity and terminates in* expected-constant *rounds; and*
- *there is a State Machine Replication (Definition 4) protocol that achieves Consistency and* expected-constant *Liveness,*

except with probability negligibly small in the security parameter.

References

[Bah13] Lear Bahack. Theoretical Bitcoin attacks with less than half of the computational power (draft). Cryptology ePrint Archive, Report 2013/868, 2013. URL: https://eprint.iacr.org/2013/868. (Cited on pages 5, 7, 17, and 18.)

[BGK+18] Christian Badertscher, Peter Gazi, Aggelos Kiayias, Alexander Russell, and Vassilis Zikas. Ouroboros genesis: Composable proof-of-stake blockchains with dynamic availability. In David Lie, Mohammad Mannan, Michael Backes, and XiaoFeng Wang, editors, *ACM CCS 2018: 25th Conference on Computer and Communications Security*, pages 913–930, Toronto, ON, Canada, October 15–19, 2018. ACM Press. https://doi.org/10.1145/3243734.3243848. (Cited on page 10.)

[BGK+21] Christian Badertscher, Peter Gazi, Aggelos Kiayias, Alexander Russell, and Vassilis Zikas. Dynamic ad hoc clock synchronization. In Anne Canteaut and François-Xavier Standaert, editors, *Advances in Cryptology – EUROCRYPT 2021, Part III*, volume 12698 of *Lecture Notes in Computer Science*, pages 399–428, Zagreb, Croatia, October 17–21, 2021. Springer, Cham, Switzerland. https://doi.org/10.1007/978-3-030-77883-5_14. (Cited on page 12.)

[BGP89] Piotr Berman, Juan A. Garay, and Kenneth J. Perry. Towards optimal distributed consensus (extended abstract). In *30th Annual Symposium on Foundations of Computer Science*, pages 410–415, Research Triangle Park, NC, USA, October 30 – November 1, 1989. IEEE Computer Society Press. https://doi.org/10.1109/SFCS.1989.63511. (Cited on page 35.)

[BKT+19] Vivek Kumar Bagaria, Sreeram Kannan, David Tse, Giulia C. Fanti, and Pramod Viswanath. Prism: Deconstructing the blockchain to approach physical limits. In Lorenzo Cavallaro, Johannes Kinder, XiaoFeng Wang, and Jonathan Katz, editors, *ACM CCS 2019: 26th Conference on Computer and Communications Security*, pages 585–602, London, UK, November 11–15, 2019. ACM Press. https://doi.org/10.1145/3319535.3363213. (Cited on pages 3 and 4.)

[BMTZ17] Christian Badertscher, Ueli Maurer, Daniel Tschudi, and Vassilis Zikas. Bitcoin as a transaction ledger: A composable treatment. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology – CRYPTO 2017, Part I*, volume 10401 of *Lecture Notes in Computer Science*, pages 324–356, Santa Barbara, CA, USA, August 20–24, 2017. Springer, Cham, Switzerland. https://doi.org/10.1007/978-3-319-63688-7_11. (Cited on page 40.)

[Can00a] Ran Canetti. Security and composition of multiparty cryptographic protocols. *Journal of Cryptology*, 13(1):143–202, January 2000. https://doi.org/10.1007/s001459910006. (Cited on page 10.)

[Can00b] Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. Cryptology ePrint Archive, Report 2000/067, 2000. URL: https://eprint.iacr.org/2000/067. (Cited on page 10.)

[CEM+20] T-H. Hubert Chan, Naomi Ephraim, Antonio Marcedone, Andrew Morgan, Rafael Pass, and Elaine Shi. Blockchain with varying number of players. Cryptology ePrint Archive, Report 2020/677, 2020. URL: https://eprint.iacr.org/2020/677. (Cited on pages 3 and 4.)

[DLP+86] Danny Dolev, Nancy A. Lynch, Shlomit S. Pinter, Eugene W. Stark, and William E. Weihl. Reaching approximate agreement in the presence of faults. *J. ACM*, 33(3):499–516, 1986. https://doi.org/10.1145/5925.5931. (Cited on pages 4, 5, and 13.)

[DP09] Devdatt P. Dubhashi and Alessandro Panconesi. *Concentration of Measure for the Analysis of Randomized Algorithms*. Cambridge University Press, 2009. https://doi.org/10.1017/CBO9780511581274. (Cited on page 38.)

[DS83] Danny Dolev and H. Raymond Strong. Authenticated algorithms for byzantine agreement. *SIAM J. Comput.*, 12(4):656–666, 1983. https://doi.org/10.1137/0212045. (Cited on page 5.)

[FGKR18] Matthias Fitzi, Peter Gaži, Aggelos Kiayias, and Alexander Russell. Parallel chains: Improving throughput and latency of blockchain protocols via parallel composition. Cryptology ePrint Archive, Report 2018/1119, 2018. URL: https://eprint.iacr.org/2018/1119. (Cited on page 3.)

[FGKR20] Matthias Fitzi, Peter Gazi, Aggelos Kiayias, and Alexander Russell. Ledger combiners for fast settlement. In Rafael Pass and Krzysztof Pietrzak, editors, *TCC 2020: 18th Theory of Cryptography Conference, Part I*, volume 12550 of *Lecture Notes in Computer Science*, pages 322–352, Durham, NC, USA, November 16–19, 2020. Springer, Cham, Switzerland. https://doi.org/10.1007/978-3-030-64375-1_12. (Cited on pages 3 and 5.)

[GKL15] Juan A. Garay, Aggelos Kiayias, and Nikos Leonardos. The Bitcoin backbone protocol: Analysis and applications. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology – EUROCRYPT 2015, Part II*, volume 9057 of *Lecture Notes in Computer Science*, pages 281–310, Sofia, Bulgaria, April 26–30, 2015. Springer Berlin Heidelberg, Germany. https://doi.org/10.1007/978-3-662-46803-6_10. (Cited on pages 3, 4, 6, 10, and 12.)

[GKL17] Juan A. Garay, Aggelos Kiayias, and Nikos Leonardos. The Bitcoin backbone protocol with chains of variable difficulty. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology – CRYPTO 2017, Part I*, volume 10401 of *Lecture Notes in Computer Science*, pages 291–323, Santa Barbara, CA, USA, August 20–24, 2017. Springer, Cham, Switzerland. https://doi.org/10.1007/978-3-319-63688-7_10. (Cited on pages 3, 4, 5, 8, 10, 16, 17, and 22.)

[GKL20] Juan Garay, Aggelos Kiayias, and Nikos Leonardos. Full analysis of nakamoto consensus in bounded-delay networks. Cryptology ePrint Archive, Report 2020/277, 2020. URL: https://eprint.iacr.org/2020/277. (Cited on pages 8, 10, 16, and 22.)

[GKL24] Juan Garay, Aggelos Kiayias, and Nikos Leonardos. The bitcoin backbone protocol: Analysis and applications. *J. ACM*, 71(4), August 2024. https://doi.org/10.1145/3653445. (Cited on page 10.)

[GKS22] Juan A. Garay, Aggelos Kiayias, and Yu Shen. Permissionless clock synchronization with public setup. In Eike Kiltz and Vinod Vaikuntanathan, editors, *TCC 2022: 20th Theory of Cryptography Conference, Part III*, volume 13749 of *Lecture Notes in Computer Science*, pages 181–211, Chicago, IL, USA, November 7–10, 2022. Springer, Cham, Switzerland. https://doi.org/10.1007/978-3-031-22368-6_7. (Cited on pages 4, 5, 6, 12, and 16.)

[GKS24] Juan A. Garay, Aggelos Kiayias, and Yu Shen. Proof-of-work-based consensus in expected-constant time. In Marc Joye and Gregor Leander, editors, *Advances in Cryptology – EUROCRYPT 2024, Part III*, volume 14653 of *Lecture Notes in Computer Science*, pages 96–125, Zurich, Switzerland, May 26–30, 2024. Springer, Cham, Switzerland. https://doi.org/10.1007/978-3-031-58734-4_4. (Cited on pages 3, 4, 6, 7, 9, 11, 12, 34, and 35.)

[GKS25] Juan A. Garay, Aggelos Kiayias, and Yu Shen. State machine replication among strangers, fast and self-sufficient. In Yael Tauman Kalai and Seny F. Kamara, editors, *Advances in Cryptology – CRYPTO 2025, Part II*, volume 16001 of *Lecture Notes in Computer Science*, pages 3–36, Santa Barbara, CA, USA, August 17–21, 2025. Springer, Cham, Switzerland. https://doi.org/10.1007/978-3-032-01878-6_1. (Cited on pages 3, 4, 5, 7, 8, 11, 12, 34, and 35.)

[GLZ25] Diana Ghinea and Chen-Da Liu-Zhang. SoK: Approximate agreement. Cryptology ePrint Archive, Report 2025/2339, 2025. URL: https://eprint.iacr.org/2025/2339. (Cited on page 5.)

[KMTZ13] Jonathan Katz, Ueli Maurer, Björn Tackmann, and Vassilis Zikas. Universally composable synchronous computation. In Amit Sahai, editor, *TCC 2013: 10th Theory of Cryptography Conference*, volume 7785 of *Lecture Notes in Computer Science*, pages 477–498, Tokyo, Japan, March 3–6, 2013. Springer Berlin Heidelberg, Germany. https://doi.org/10.1007/978-3-642-36594-2_27. (Cited on pages 10 and 39.)

[LL22] Christoph Lenzen and Julian Loss. Optimal clock synchronization with signatures. In Alessia Milani and Philipp Woelfel, editors, *41st ACM Symposium Annual on Principles of Distributed Computing*, pages 440–449, Salerno, Italy, July 25–29, 2022. Association for Computing Machinery. https://doi.org/10.1145/3519270.3538444. (Cited on page 5.)

[McD98] Colin McDiarmid. Concentration. In Michel Habib, Colin McDiarmid, Jorge Ramirez-Alfonsin, and Bruce Reed, editors, *Probabilistic Methods for Algorithmic Discrete Mathematics*, pages 195–248, Berlin, Heidelberg, 1998. Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-662-12788-9_6. (Cited on page 39.)

[Nak08] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system, Dec 2008. URL: https://bitcoin.org/bitcoin.pdf. (Cited on page 3.)

[PS17] Rafael Pass and Elaine Shi. FruitChains: A fair blockchain. In Elad Michael Schiller and Alexander A. Schwarzmann, editors, *36th ACM Symposium Annual on Principles of Distributed Computing*, pages 315–324, Washington, DC, USA, July 25–27, 2017. Association for Computing Machinery. https://doi.org/10.1145/3087801.3087809. (Cited on page 12.)

[PSs17] Rafael Pass, Lior Seeman, and abhi shelat. Analysis of the blockchain protocol in asynchronous networks. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *Advances in Cryptology – EUROCRYPT 2017, Part II*, volume 10211 of *Lecture Notes in Computer Science*, pages 643–673, Paris, France, April 30 – May 4, 2017. Springer, Cham, Switzerland. https://doi.org/10.1007/978-3-319-56614-6_22. (Cited on page 10.)

[YNHS20] Haifeng Yu, Ivica Nikolic, Ruomu Hou, and Prateek Saxena. OHIE: Blockchain scaling made simple. In *2020 IEEE Symposium on Security and Privacy*, pages 90–105, San Francisco, CA, USA, May 18–21, 2020. IEEE Computer Society Press. https://doi.org/10.1109/SP40000.2020.00008. (Cited on page 3.)

A Mathematical Facts

We assuming finite probability spaces and random variables with finite means.

Theorem 20 (Chernoff bounds). *Suppose* $\{X_i : i \in [n]\}$ *are mutually independent Boolean random variables, with* $\mathbf{Pr}[X_i = 1] = p$*, for all* $i \in [n]$*. Let* $X = \sum_{i=1}^{n} X_i$ *and* $\mu = pn$*. Then, for any* $\delta \in (0, 1]$*, it holds that*

$$\mathbf{Pr}[X \leq (1-\delta)\mu] \leq e^{-\delta^2\mu/2} \text{ and } \mathbf{Pr}[X \geq (1+\delta)\mu] \leq e^{-\delta^2\mu/3}.$$

Also, for all $t > 0$*,*

$$\mathbf{Pr}[X \geq \mu + t] \leq e^{-2t^2 n}.$$

Definition 6. [DP09, Definition 5.3] *A sequence of random variables* $(X_0, X_1, \ldots)$ *is a martingale with respect to the sequence* $(Y_0, Y_1, \ldots)$*, if, for all* $n \geq 0$*,* X_n *is determined by* $Y_0, \ldots, Y_n$ *and* $\mathbb{E}[X_{n+1} \mid Y_0, \ldots, Y_n] = X_n$*.*

Theorem 21. [McD98, Theorem 3.15] *Let $(X_0, X_1, \ldots)$ be a martingale with respect to the sequence $(Y_0, Y_1, \ldots)$. Suppose an event G implies*

$$X_k - X_{k-1} \leq b \text{ (for all } k) \text{ and } V = \sum_k \mathbf{Var}[X_k - X_{k-1} \mid Y_1, \ldots, Y_{k-1}] \leq v.$$

Then, for non-negative n and t,

$$\mathbf{Pr}[X_n \geq X_0 + t \wedge G] \leq \exp\Big\{ - \frac{t^2}{2v + 2bt/3} \Big\}.$$

Lemma 22. *When $|r| \leq 1$, $\lim_{k=0}^{\infty} ar^k = a/(1-r)$.*

B Preliminaries (Cont'd)

Global clock. We adopt $\mathcal{G}_{\mathsf{Clock}}$ (cf. [KMTZ13]) to model perfectly-synchronous clocks.

Functionality $\mathcal{G}_{\mathsf{Clock}}$

This functionality maintains state variables as follows.

State Variable	Description
$\mathcal{P} \leftarrow \emptyset$	The set of registered parties $\mathsf{P} = (\text{pid}, \text{sid})$.
$F \leftarrow \emptyset$	The set of registered functionalities (together with their session identifier).
$\tau_{\text{sid}} \leftarrow 0$	The clock variable for session sid.
$d_{\mathsf{P}} \leftarrow 0$	The clock-update variable for $\mathsf{P} = (\text{pid}, \text{sid}) \in \mathcal{P}$.
$d(\mathcal{F}, \text{sid}) \leftarrow 0$	The clock-update variable for $(\mathcal{F}, \text{sid}) \in F$.

- Upon receiving $(\text{CLOCK-UPDATE}, \text{sid}_C)$ from some party $\mathsf{P} \in \mathcal{P}$ set $d_{\mathsf{P}} \leftarrow 1$; execute *Round-Update* and forward $(\text{CLOCK-UPDATE}, \text{sid}_C, \mathsf{P})$ to $\mathcal{A}$.
- Upon receiving $(\text{CLOCK-UPDATE}, \text{sid}_C)$ from some functionality $\mathcal{F}$ in a session sid such that $(\mathcal{F}, \text{sid}) \in F$ set $d(\mathcal{F}, \text{sid}) \leftarrow 1$, execute *Round-Update* and return $(\text{CLOCK-UPDATE}, \text{sid}_C, \mathcal{F})$ to this instance of $\mathcal{F}$.
- Upon receiving $(\text{CLOCK-READ}, \text{sid}_C)$ from any participant (including the environment on behalf of a party, the adversary, or any ideal—shared or local—functionality) return $(\text{CLOCK-READ}, sid_C, \tau_{\text{sid}})$ to the requestor (where sid is the sid of the calling instance).

Procedure Round-Update: For each session sid do: If $d(\mathcal{F}, \text{sid}) = 1$ for all $\mathcal{F} \in F$ and $d_{\mathsf{P}} = 1$ for all honest partyset $P = (\cdot, \text{sid}) \in \mathcal{P}$, then set $\tau_{\text{sid}} \leftarrow \tau_{\text{sid}} + 1$ and reset $d(\mathcal{F}, \text{sid}) \leftarrow 0$ and $d_{\mathsf{P}} \leftarrow 0$ for all partyset $\mathsf{P} = (\cdot, \text{sid}) \in \mathcal{P}$.

Functionality 1: *The global clock.*

Random oracle. By convention, we model the hash function H used to generate PoW as a random oracle $\mathcal{F}_{\mathsf{RO}}$. Note that, we need additionally a wrapper functionality that wraps $\mathcal{F}_{\mathsf{RO}}$ to capture the restrictions on adversarial access (thus fulfilling the honest-majority assumption). For brevity, we omit such considerations here and refer to [BMTZ17] for more technical details.

Functionality $\mathcal{F}_{\mathsf{RO}}$

The functionality is parameterized by a security parameter κ.

State Variable	Description
$\mathcal{P} \leftarrow \emptyset$	The set of registered parties.
$H \leftarrow \emptyset$	A dynamically updatable function table where $H[x] = \perp$ denotes the fact that no pair of the form $(x, \cdot)$ is in H.

- **Eval.** Upon receiving $(\text{EVAL}, \text{sid}, x)$ from some party $\mathsf{P} \in \mathcal{P}$ (or from $\mathcal{A}$ on behalf of a corrupted P), do the following:
  1. If $H[x] = \perp$ sample a value y uniformly at random from $\{0,1\}^\kappa$ and set $H[x] \leftarrow y$.
  2. Return $(\text{EVAL}, \text{sid}, x, H[x])$ to the requestor.

Functionality 2: *The random oracle.*

Diffuse network. We model the synchronous communication by $\mathcal{F}_{\mathsf{Diffuse}}$ [BMTZ17].

Functionality $\mathcal{F}_{\mathsf{Diffuse}}$

The functionality is parameterized by the network delay Δ.

State Variable	Description
$\mathcal{P} \leftarrow \emptyset$	The set of registered parties.
$\vec{M} \leftarrow []$	A dynamically updatable list of quadruples $(m, \mathsf{mid}, D_{\mathsf{mid}}, \mathsf{P})$ where D_{mid} denotes the fetch counter.

Network capabilities:

- Upon receiving $(\text{DIFFUSE}, \text{sid}, m)$ from some $\mathsf{P}_s \in \mathcal{P}$, where $\mathcal{P} = \{\mathsf{P}_1, \ldots, \mathsf{P}_n\}$ denotes the current party set, do:
  1. Choose n new unique message-IDs $\mathsf{mid}_1, \ldots, \mathsf{mid}_n$.
  2. Initialize $2n$ new variables $D_{\mathsf{mid}_1} := D^{MAX}_{\mathsf{mid}_1} \ldots := D_{\mathsf{mid}_n} := D^{MAX}_{\mathsf{mid}_n} := 1$ and a per message-delay $\Delta_{\mathsf{mid}_i} = \Delta$ for $i \in [n]$.
  3. Set $\vec{M} := \vec{M} \parallel (m, \mathsf{mid}_1, D_{\mathsf{mid}_1}, \mathsf{P}_1) \parallel \ldots \parallel (m, \mathsf{mid}_n, D_{\mathsf{mid}_n}, \mathsf{P}_n)$.
  4. Send $(\text{DIFFUSE}, \text{sid}, m, \mathsf{P}_s, (\mathsf{P}_1, \mathsf{mid}_1), \ldots, (\mathsf{P}_n, \mathsf{mid}_n))$ to the adversary.
- Upon receiving $(\text{FETCH}, \text{sid})$ from $\mathsf{P} \in \mathcal{P}$, or from $\mathcal{A}$ on behalf of a corrupted party P:
  1. For all tuples $(m, \mathsf{mid}, D_{\mathsf{mid}}, \mathsf{P}) \in \vec{M}$, set $D_{\mathsf{mid}} := D_{\mathsf{mid}} - 1$.
  2. Let $\vec{M}^{\mathsf{P}}_0$ denote the subvector $\vec{M}$ including all tuples of the form $(m, \mathsf{mid}, D_{\mathsf{mid}}, \mathsf{P})$ with $D_{\mathsf{mid}} \leq 0$ (in the same order as they appear in $\vec{M}$). Delete all entries in $\vec{M}^{\mathsf{P}}_0$ from $\vec{M}$

and in case some $(m, \mathsf{mid}, D_{\mathsf{mid}}, \mathsf{P})$ is in $\vec{M}_0^{\mathsf{P}}$, where P is honest, set $\Delta_{\mathsf{mid}'} = \Delta$ for any $(m, \mathsf{mid}', D_{\mathsf{mid}'}, (\cdot, \mathrm{sid}))$ in $\vec{M}$ and replace this record by $(m, \mathsf{mid}', \min\{D_{\mathsf{mid}'}, \Delta\}, \mathsf{P}')$.

3. Output $\vec{M}_0^{\mathsf{P}}$ to P (if P is corrupted, send $\vec{M}_0^{\mathsf{P}}$ to $\mathcal{A}$).

Additional adversarial capabilities:

- Upon receiving $(\textsc{diffuse}, \mathrm{sid}, m)$ from some corrupted $\mathsf{P}_s \in \mathcal{P}$ (or from $\mathcal{A}$ on behalf of P_s if corrupted), execute it the same way as an honest-sender diffuse, with the only difference that $\Delta_{\mathsf{mid}_i} = \infty$.
- Upon receiving $(\textsc{delays}, \mathrm{sid}, (T_{\mathsf{mid}_{i_1}}, \mathsf{mid}_{i_1}), \ldots, (T_{\mathsf{mid}_{i_\ell}}, \mathsf{mid}_{i_\ell}))$ from the adversary do the following for each pair $(T_{\mathsf{mid}_{i_j}}, \mathsf{mid}_{i_j})$: if $D^{MAX}_{\mathsf{mid}_{i_j}} + T_{\mathsf{mid}_{i_j}} \leq \Delta_{\mathsf{mid}_{i_j}}$ and mid_{i_j} is a message-ID of receiver $\mathsf{P} = (\cdot, \mathrm{sid})$ registered in the current $\vec{M}$, set $D_{\mathsf{mid}_{i_j}} := D_{\mathsf{mid}_{i_j}} + T_{\mathsf{mid}_{i_j}}$ and set $D^{MAX}_{\mathsf{mid}_{i_j}} := D^{MAX}_{\mathsf{mid}_{i_j}} + T_{\mathsf{mid}_{i_j}}$; otherwise, ignore this pair.
- Upon receiving $(\textsc{swap}, \mathrm{sid}, \mathsf{mid}, \mathsf{mid}')$ from the adversary, if mid and mid' are message-IDs registered in the current $\vec{M}$, then swap the triples $(m, \mathsf{mid}, D_{\mathsf{mid}}, (\cdot, \mathrm{sid}))$ and $(m, \mathsf{mid}', D_{\mathsf{mid}'}, (\cdot, \mathrm{sid}))$ in $\vec{M}$. Return $(\textsc{swap}, \mathrm{sid})$ to the adversary.
- Upon receiving $(\textsc{get-reg}, \mathrm{sid})$ from $\mathcal{A}$, return the response $(\textsc{get-reg}, \mathrm{sid}, \mathcal{P})$ to $\mathcal{A}$.

Functionality 3: *The diffusion network.*

Public setup. We model the public setup by a CRS functionality $\mathcal{F}_{\mathsf{CRS}}$.

Functionality $\mathcal{F}_{\mathsf{CRS}}$

The functionality is parameterized by a distribution $\mathcal{D}$.

- **Retrieve.** Upon receiving $(\textsc{Retrieve}, \mathrm{sid})$ from some party P (or from $\mathcal{A}$ on behalf of a corrupted P), do the following:
  1. If activated for the first time, choose a value $d \leftarrow \mathcal{D}$, and send $(\textsc{Retrieved}, \mathrm{sid})$ to $\mathcal{F}_{\mathsf{RO}}$ and $\mathcal{F}_{\mathsf{Diffuse}}$.
  2. Return $(\textsc{Retrieve}, d)$ to P.

Functionality 4: *The common reference string.*

C Glossary

In Table 2, we list all the protocol parameters and their explanation.

Variable	Description
κ	Security parameter; length of the random oracle output.
m	The number of parallel chains ($m = \Theta(\log^2 \kappa)$).

M	The length of a target recalculation epoch in number of rounds; each epoch has an input phase of M_{IB} rounds, which begins after M_{FR} rounds after the onset of the epoch and ends before M_{CP} rounds before the conclusion of the epoch.
T	The target to successfully solve a PoW.
h_r	Number of honest parties at round r.
t_r	Number of corrupted parties at round r.
δ	Advantage of honest parties ($t \leq (1-\delta)h$).
f	Initial (and ideal) block generation rate.
ϵ	Ideal quality of concentration of random variables.

Table 2: *Main parameters of the protocol.*