Consensus Redux: Distributed Ledgers in the Face of Adversarial Supremacy

Christian Badertscher[⋆], Peter Gaži[⋆⋆], Aggelos Kiayias[⋆⋆⋆], Alexander Russell[†], and Vassilis Zikas[‡]

August 24, 2020

Abstract. Distributed ledgers, such as those arising from blockchain protocols, have been touted as the centerpiece of an upcoming security-critical information technology infrastructure. Their basic properties—consistency and liveness—can be guaranteed under specific constraints about the resources of an adversary relative to the resources of the nodes that follow the protocol. Given the intended long-livedness of these protocols, perhaps the most fundamental open security question currently is their behavior and potential resilience to temporary spikes in adversarial resources.

In this work we give the first thorough treatment of self-healing properties of distributed ledgers covering both proof-of-work (PoW) and proof-of-stake (PoS) protocols. Our results quantify the vulnerability period that corresponds to an adversarial spike and classify three types of currently deployed protocols with respect to their self-healing ability: PoW-based blockchains, PoS-based blockchains, and iterated Byzantine Fault Tolerant (iBFT) protocols.

1 Introduction

Consensus in the presence of dishonest majority is impossible.[1] This fundamental mathematical fact permeates the space of distributed ledgers and suggests the optimality of protocols that come close to the 50% threshold. It is worth emphasizing that even reaching this bound requires bounded network delays; otherwise, things are even worse: the adversary must be restricted to below 1/3, [13]. The Bitcoin blockchain protocol [29] essentially achieves this 50% bound in a particularly adverse setting where, among other challenges, users fluctuate over time [19,33,20].

Unfortunately, these mathematical impossibility results do nothing to rule out the viability of dishonest majority attacks against real world consensus protocols. So what can a dishonest majority attacker do? A distributed ledger, a concept popularized by Nakamoto's blockchain protocol, aims at realizing and maintaining a transaction log by an ever-evolving population of maintainers that continuously collect transactions and incorporate them in a certain admissible way to the log. Distributed ledgers possess two fundamental properties, *consistency* and *liveness*, and both are susceptible to a dishonest majority attack. In the special case of a ledger carrying a cryptocurrency, a consistency violation can permit an attacker to mount a "double spending" attack; a liveness violation, on the other hand, can permit an attacker to censor specific transactions.

Do such attacks pose a real threat against distributed ledgers? While liveness attacks are, by nature, difficult to detect, consistency violations have been extensively documented against various cryptocurrencies that are based on blockchain protocols; these have had devastating consequences. Some examples include

[⋆] IOHK, `christian.badertscher@iohk.io`. Work partially done while author was at the University of Edinburgh.

[⋆⋆] IOHK, `peter.gazi@iohk.io`.

[⋆⋆⋆] University of Edinburgh and IOHK. `akiayias@inf.ed.ac.uk`. Research partly supported by EU Project No. 780477, PRIVILEDGE.

[†] University of Connecticut and IOHK. `acr@cse.uconn.edu`. This material is based upon work supported by the National Science Foundation under Grant No. 1717432.

[‡] University of Edinburgh and IOHK. `vassilis.zikas@ed.ac.uk`.

[1] Note that consensus is a different problem to that of "reliable broadcast" that is feasible for any number of malicious users. For a formal proof of the impossibility for the classical consensus problem we refer to [16]. For a formal proof of the impossibility for the cae of distributed ledgers we refer to [18].

Horizen (formerly known as ZenCash) [41], Vertcoin [36], Bitcoin Gold [21], and Ethereum Classic [30]; the reader is referred to [6] for further discussion.

While a dishonest majority attack can be contemplated for any consensus scheme, open distributed ledgers, as the above examples suggest, can be particularly susceptible to dishonest majority attacks. Building on concepts such as proof-of-work (PoW) (e.g., Nakamoto's Bitcoin protocol [29] or Ethereum [7]) or proof-of-stake (PoS) (e.g., Ouroboros [23] or Algorand [9]), these protocols are susceptible to the sudden adversarial coordination of significant computational power or the sudden acquisition of substantial stake that can be harnessed for an attack. In fact, in some cases users can anticipate a dishonest majority attack! For instance, in the attack on Ethereum Classic mentioned above, the cryptocurrency experienced a significant drop (by about 50%[2]) in its PoW difficulty some few months before the attack was mounted, thus making it an "easy" target in the view of an attacker. A different example is the case of Bitcoin Cash and Bitcoin SV, which entered a into a "hash war" [31] due to differences in the core Bitcoin implementation.

The above highlight what is perhaps the most fundamental unresolved question about the security of distributed ledgers. What is the exact impact of dishonest majority attacks? Once the "spike" of adversarial supremacy is over, will a distributed ledger return to normalcy? In such a case, how long will this take and what are the precautions that users may need to take if they anticipate such an attack?

1.1 Our Results

We present the first complete treatment of distributed ledger security in the presence of over 50% spikes in terms of adversarial access to resources. Our model and analysis applies to both PoW and PoS protocols. We use our framework to analyze a number of protocols, and quantify their susceptibility to dishonest majority attacks as well as their ability to recover, which is known under the term "self-healing." We describe our results in more detail.

Self-healing distributed ledgers. Blockchain protocols are complex cryptographic schemes that rely on an array of different underlying primitives for ensuring fairness and security, including PoW and PoS; analyzing them with a single lens, in a setting of adversity for which they were not even designed, requires special care. In order to address this challenge, we put forth a novel protocol abstraction layer that captures the salient protocol features that are relevant in a setting of a spike in adversarial resources.

Given our abstraction, we can model any attack against the fundamental properties of the ledger as a "settlement game" between an attacker and a challenger that proceeds as follows. In each protocol step, the attacker can provide a participation instruction that determines the number of honest and adversarial parties that are online for that step. Subsequently, the challenger receives this instruction and uses a protocol dependent distribution to determine an outcome for the lottery that abstracts the PoW or PoS element of the underlying protocol. For instance, in a PoW-based blockchain, the lottery determines the PoW solutions that were found by the online parties. In a PoS-based blockchain, it determines which parties are capable of issuing a PoS for that step. The challenger provides the lottery outcome to the adversary who is tasked to consistently advance the protocol execution. The execution itself is modeled by a directed tree that is suitably labeled by the adversary to indicate the parties that are active in each step of the protocol. Given such labeling, the ledger of each party can be abstracted as a suitable minor of the execution graph. We note that the labeling and the projection to the minor is subject to protocol-dependent logic, e.g., in the case of PoW-based protocols, parties always follow the most difficult chain. The adversary wins the settlement game if it activates either a consensus violation or a liveness violation with respect to specific protocol steps. A secure protocol ensures that such violations do not happen for any protocol step in the whole execution.

With the above abstraction we are in a position to discuss the self-healing property of a distributed ledger. Under honest majority conditions, the participation instructions given by the adversary in the settlement game are subject to the condition that honest parties' resources should suitably exceed those of adversarial parties. To capture an adversarial spike we can specify an excess $\mathcal{B}$ and allow the adversary to violate the honest majority assumption by strategically spending her budget with participation instructions that favor

[2] See `https://bitinfocharts.com/comparison/difficulty-etc.html`.

the adversarial parties. The spike of adversity has a beginning and an end, delineated by the onset of the honest majority violation and the step that spends the final portion of $\mathcal{B}$. The question we then ask for such a settlement game is how liveness and consistency violations can be controlled as a function of the adversarial budget $\mathcal{B}$ and the positioning of the stake, i.e., determining the *vulnerability period* as a function of the spike.

Analysis of Nakamoto's blockchain protocol. We show that the Nakamoto PoW blockchain achieves self-healing. First, we provide an asymptotic treatment: we show that, roughly speaking, if the protocol is under an adversarial-majority attack with budget $\mathcal{B}$ in a time interval $[t_a, t_b]$, then standard honest-majority persistence and liveness guarantees (as established by previous works) still hold outside of the interval $[t_a - O(\mathcal{B}), t_b + O(\mathcal{B})]$. In other words, a self-healing period of length linear in the adversarial attack budget is sufficient for the protocol guarantees to return to normalcy; and moreover, the honest-majority persistence and liveness guarantees are maintained also sufficiently prior to the spike. To give this theoretical analysis practical relevance, we also give concrete instantiations of our asymptotic bounds and present exact recovery times for several illustrative settings.

Analysis of Nakamoto-style PoS. We then turn our attention to Nakamoto-style PoS protocols, and choose Ouroboros Genesis [2] as a representative of this class; Ouroboros is deployed in the Cardano cryptocurrency.[3] We again formally prove that Ouroboros Genesis achieves self-healing; intriguingly, this happens at the same asymptotic rate as in the PoW case above. This is surprising in the context of previous results showing that the violation probability of the closely related κ-common prefix property in the honest-majority setting with honest advantage ε appears to scale quite differently with respect to ε: it follows $\exp(-\varepsilon^2 \Omega(\kappa))$ in the PoW case, whereas only $\exp(-\varepsilon^3 \Omega(\kappa))$ is known for PoS.

There is nonetheless one aspect in which our result for Nakamoto-style PoS is weaker than the corresponding result for PoW: When Nakamoto-style PoS execution is divided into epochs, in order to accomodate shifts in stake between participants, adversarial spikes' maximum length is linearly dependent on the length of the epoch.

Analysis of iterated BFT PoS. We finally model and analyze "iterated BFT" PoS protocols using our ledger abstraction, choosing Algorand as described and analyzed in [9] to be the representative of this class. Such PoS protocols iterate a basic BFT consensus step (e.g., such as [13,15,8]) using some cryptographic mechanism to map the full population of stakeholders to the set of active participants for a certain protocol step. Unsurprisingly, the result of our analysis in this case is a negative one: an adversarial spike above the known resilience bound, even a small one, leads to an unbounded vulnerability period as the protocol does not self-heal.[4]

1.2 Related Work

Abstracting the execution of a blockchain protocol as a probability distribution over a family of directed graphs is an approach that has been quite frequently utilized in previous works, notably the analysis of selfish-mining [14,39] and mining games [22] in the PoW setting and the Ouroboros protocols in the PoS setting [23,11]. A key benefit of the approach is that high level protocol properties (e.g., the consistency of the ledger or the fairness of rewards) can be expressed as graph properties defined over the support of such distributions and subsequently the density of events relevant for these properties can be measured in the resulting probability distributions. Still, none of these previous works offered a model sufficiently detailed to capture all the relevant consensus properties (persistence and liveness) within a general threat model where parties may adversarially go offline. At the same time none of the previous works was sufficiently expressive

[3] `http://cardano.org`

[4] We remark that in [9, Section 10], it is mentioned that a "fork resolution" mechanism is possible in the context of Algorand and it will be provided in some future version of the protocol. This points to the fact that a hybridization between iterated-BFT and Nakamoto-style design elements is possible and, from the perspective of our work, it might be possible to take advantage of this and demonstrate more favorable self-healing properties for such hybrid protocols.

to be applicable to any consensus protocol be it longest-chain-based, graph-based or iterated BFT-based, as we do here with our settlement game abstraction.

Several past works studied self-healing algorithms, e.g., [38,32], but Byzantine faults have received less attention, see e.g., [26]. To the best of our knowledge, the only previous works that investigate security of blockchain protocols under temporary adversarial majority is by Avarikioti *et al.* [1] and Bentov *et al.* [4]. The former work focuses on Bitcoin and hence on the proof-of-work case. Their model is significantly more restricted in the sense that (i) instead of the adversary adaptively determining participation as in our settlement game, it issues sleeping instructions that are adhered to with a certain probability of success based on independent coin flips; (ii) the impact of the spike is not quantified in the time-domain with respect to self-healing; our results instead show when the protocol's behavior will return to normality as a function of the spike. The latter work cited above considers self-healing for Spacemesh [4], a consensus protocol based on Proof of SpaceTime. Compared to Bitcoin and the results we obtain in this work, their self-healing guarantee relies crucially on the existence of a period of honest participation above a certain threshold and a temporary adversarial minority controlling $\ll 1/3$ of the spacetime resources during this period to reach persistence and liveness eventually after an adverse situation.

2 Underlying Model

The execution environment of Nakamoto-style consensus differs from traditional distributed computing environments in several respects: First, the system allows parties to arbitrarily join and leave the protocol; specifically, the protocol does not require knowledge of which—or even how many—parties are participating. Likewise, the system supports temporary unavailability of parties, e.g., due to network shortage or even a conscious decision to neglect their "protocol-duties." Such temporary lack of availability was first captured as "sleepiness" in [34] and then extended to a fully granular setting in the dynamic availability framework from [2] building on the more coarse-grained treatment in [3].

Adversarial behavior. Cryptographic analyses of Nakamoto style blockchain ledgers consider at least two types of parties: honest and corrupted. Honest parties are guaranteed to execute the protocol specification, whereas corrupted parties are assumed to be controlled by an active adversary who might make them misbehave in order to violate some of the protocol's security guarantees (see below).

Network, time, and (partial) synchrony. Consistent with prior work performing cryptographic analysis of PoW and PoS blockchains [19,33,3,2,10] we adopt a discretized view of protocol time. Concretely, we assume that the protocol advances in rounds, where every party is aware of the current round index. Formally, this can be captured as in [3,2] by assuming access to a global clock. We will assume that the messages are circulated by means of a reliable[5] diffusion network with an upper bound Δ on the number of rounds it takes for a message sent by an honest party to be delivered to other honest parties; we allow the adversary to deliver any such message m with different delays $\Delta_{m,i} \leq \Delta$ to each honest party p_i. As shown in [3,2], this can be captured by means of a *delayed multicast network functionality.*

3 Abstracting Ledger Protocol Executions

Ledger protocols are complex objects; analyzing them typically requires accounting for the peculiarities of each specific protocol and its underlying assumptions. Several frameworks have beed suggested for such analysis which focus on different aspects of the protocol—e.g., laying down basic properties, incorporating incentives, allowing dynamic participation, and composability to name some. However, these frameworks are insufficient for addressing the security of ledger protocols in the presence of (even temporary periods of) adversarial majority.

In this section we introduce a novel abstraction of distributed ledger protocols and an accompanying definitional framework. As demonstrated in the following sections, our framework can be used, instead of previous frameworks, for analyzing the core consensus properties of common ledger protocols. But what

[5] By this we mean that honest party messages are eventually delivered to all parties.

distinguishes our framework from existing models, is its ability of our abstraction to address *self-healing properties*, i.e., their return to normalcy after a period of adversarial majority— also referred to as a *(corruption) spike*—in the dynamic participation setting. Furthermore, our framework is the first framework that is general enough to address self-healing both for Nakamoto-style and for iterated Byzantine Fault-Tolerant (iBTF) protocols (see below).

Mainstream approaches to decentralized ledgers rely on the following high level idea: The protocol implements a lottery, weighted according to a particular resource, to define for each round a collection of distinguished parties (or rather addresses/identifiers). For example, in the case of Bitcoin the underlying resource is the hashing power enabling Proofs of Work (PoW); in the case of Ouroboros and Algorand it is the virtual resource stake, enabling Proofs of Stake (PoS). While the exact mechanism that realizes this implicit lottery is slightly different, the essential properties necessary for proving security of the protocol's consensus layer are very similar.

Nakamoto-style Proof-of-Work (PoW): In the first class of protocols, which includes Bitcoin, this implicit lottery is realized by the parties repeatedly hashing randomly selected potential solutions until a correct solution to a "hashing puzzle" is discovered. Looking for a solution can be modelled as Bernoulli trials with winning probability that depends on the difficulty of the puzzle. As a result, depending on the participation in the protocol, this process might result in zero, one, or more parties, often called "miners" in this context, winning each round's lottery. And since both the honest parties and the adversarially controlled parties might be mining, either (or both) of them might win in a particular round.

Nakamoto-style Proof-of-Stake (PoS): In the second class of protocols, which includes Ouroboros family of protocols, the situation is similar to the above, with the main difference being that, by design, parties win the implicit lottery with probability proportional to their existing stake, i.e., the total fraction of the coins that the party owns.

Iterated-BFT with Proof-of-Stake (iBFT): Finally, a third class of protocols, which includes Algorand, utilizes a similar lottery as above in a slightly different way, namely it repeatedly elects sets of parties, so called *committees*, that are in charge of proposing and subsequently endorsing the next block of a chain. Participation as either a proposer or an endorser is stake-based and weighted accordingly as in the case of Nakamoto-style PoS.

In a nutshell, in Nakamoto-style protocols the above lottery assigns to each round[6] zero, one or several lottery leaders that are charged with proposing the next block; in iBFT, the lottery might output two types of winners: block proposers and block verifiers who are charged with voting—for example a threshold-vote mechanism such as majority vote—on which of the currently proposed blocks (when more than one block proposal exists) should be included as the next block.

The ledger protocol abstraction. In the following, we distill the common structure of the standard blockchain protocols discussed above and introduce the notion of an *abstraction* of a ledger protocol. We use Nakamoto-style PoW ledger protocols as our running example to motivate the different components of our abstraction. The abstraction captures in an accurate manner the common features of various ledger protocols and provides us the language for introducing a cryptographic game which will be used to define different security properties, in particular related to self-healing as discussed above. Importantly, the abstraction can be used for all three types of ledger protocols discussed above, and leaves lower level, protocol-dependent features to be further defined whenever needed in the analysis. Such lower level instantiations of the general abstraction are then introduced in the following sections when the corresponding protocols are analyzed.

A *ledger (protocol) abstraction* is a tuple $(\mathbb{P}, \Sigma, \mathcal{W}, \mathcal{D}, \mathcal{G}, \dot{\rightarrow}, \mathbb{L})$ specified as follows:

- The set $\mathbb{P}$ is the universe of possible party identities.

[6] In some previous works, the term "slots" is often used instead of "rounds"; here we stick to the round terminology which is used in a wider range of works.

- Σ is an alphabet that specifies *participation indication* at any given round. In a nutshell, in any execution of the ledger protocol, a symbol $\sigma \in \Sigma$ is associated to each round[7] which indicates the level of participation for both honest parties and the adversary. Note that participation does not necessarily count physical parties or identities but can be expressed in units of the resource underlying the protocol security. For instance, as we shall see below, in Nakamoto-style PoW, a σ associated with a given round will encode the number of hash queries performed by honest parties and the number of hash queries performed by corrupted parties in the above round. Looking ahead, in order to define a worst-case adversary's winning conditions in a self-healing game (see Figure 1) we will allow the participation indicator to be chosen by the adversary for any round.
- The set $\mathcal{W}$ is the support of the distribution induced by the lottery corresponding to the ledger protocol. $\mathcal{W}$ is protocol dependent. For instance, in the Nakamoto-style PoW case, w will be the actual number of honest and/or corrupted parties that solved the hash puzzle of the current round (i.e., that won the lottery defined by this puzzle).
- $\mathcal{D}$ is a sampler that captures the output of the above-discussed lottery as a function of the participation. Concretely, given a participation indicator $\sigma \in \Sigma$, $\mathcal{D}(\sigma)$ produces an outcome in $\mathcal{W}$; informally, each $\mathcal{D}(\sigma)$ defines a random variable corresponding to the outcome of the respective lottery associated with the current round. A sequence of samples $w_1, \ldots, w_m$ from $\mathcal{D}(\cdot)$ will be called a *characteristic string* (and, as we see below, will be associated to a particular protocol execution).

  For instance, in the Nakamoto-style PoW case $\mathcal{D}$ will be sampling each w (as in the previous item) by appropriate Bernoulli trials parametrized by the participation indicators in w (number of hashing queries).
- $\mathcal{G}$ is a set of (annotated) *execution graphs.* Informally, an execution graph associated to any given protocol round is a snapshot of all security-relevant events in the protocol execution, i.e., each node of the execution graph corresponds to an event of interest during the actual protocol execution. In our work, execution graphs of a specific form will be sufficient to capture the relevant events of all three types of ledger protocols treated. As a general pattern, all execution graphs will include (at least) nodes corresponding to block creation by a slot leader and to the outcome of the corresponding lotteries. Each $G \in \mathcal{G}$ will be a rooted tree equipped with two labelling functions $\mathsf{l}_{\#}$ and $\mathsf{l}_{\mathsf{meta}}$. To each node of G, $\mathsf{l}_{\#}$ assigns a positive integer—the round that the event recoded in this node occurred— and $\mathsf{l}_{\mathsf{meta}}$ assigns a metadata-label over $\mathcal{W} \times \{\mathsf{h}, \mathsf{a}\} \times 2^{\mathbb{P}} \times \mathrm{AUX}$, where $\mathrm{AUX} \in \{0,1\}^* \cup \{\bot\}$. In particular, $\mathsf{l}_{\mathsf{meta}}(v)$ will annotate v with the characteristic string w of that round, the information of whether this round can be though of as "honest" or "corrupted" (e.g., whether a unique honest party won the lottery in this round), the set of parties that have observed the sequence of events in the path from the root r to v, and (possibly) some additional protocol-dependent auxiliary information.

  The following labelling conventions are common in the graphs G of all protocols considered here: (i) The sequence of labels $\mathsf{l}_{\#}()$ is non-decreasing along any path of the tree G, with the root r labeled as $\mathsf{l}_{\#}(r) = 0$. (ii) $\mathsf{l}_{\mathsf{meta}}$ might assign any particular label $P \in \mathbb{P}$ to at most one vertex of G, reflecting the fact that a party P can be observing a specific event at a point in time.
- The relation $\overset{w}{\to}$: Since an execution graph G encodes security-relevant events in a protocol execution, as the ledger protocol evolves, the execution graph changes. Thus each execution of L rounds of any ledger protocol, is characterized by a sequence $G_0, G_1, \ldots, G_L$ of execution graphs. Note that this sequence is not necessarily monotone with respect to the subgraph-relation, as one does not need to always keep track of all past events. Instead, the protocol defines for any given $w \in \mathcal{W}$ a relation $\overset{w}{\to}$ over $\mathcal{G}$, which expresses a *valid*, according to w, transition between execution graphs.

  or the three protocol types discussed here $\overset{w}{\to}$ will be defined as follows: $G \overset{w}{\to} G'$, if the following conditions hold:[8] (i) G can be derived from G' by iteratively deleting leaf nodes and their associated incoming edges, where all labels must be preserved except for the parties that $\mathsf{l}_{\mathsf{meta}}$ has assigned; intuitively, this property captures the fact that G must correspond to a proper history of G', while only parties change their view

[7] As we shall see, this is extracted by the adversary's behavior and the actual participation levels in each round.

[8] This is the set of conditions that apply to all the protocol types considered here. Additional protocol-dependent conditions about the placement of the $\mathbb{P}$ labels and the way G' extends G might be incurred by different protocols.

from G to G'. (ii) $\mathsf{l}_{\mathsf{meta}}$ assigns a label $(w, \cdot, \cdot, \cdot)$ to all nodes which are in G' and not in G; this captures the fact that the transition from G to G' is properly annotated as occurring through w.

- The operation $\mathbb{L}$: Any distributed ledger protocol explicitly defines a way of extracting the settled part of the ledger (also called the ledger's state) from the view of any honest party. In blockchain-based ledger protocols, such as the ones considered here, this is the (prefix of) the current blockchain seen by this party which is guaranteed (except with very small probability) to be eventually agreed upon by everyone. In particular, for any given $P \in \mathbb{P}$ such that P is assigned to some node in G, the operator $\mathbb{L}$ will extract from G the path that includes the ordered sequence of all block-creation events corresponding to the above settled chain. (Recall that our execution graphs include a block-creation event for any block every created by an honest party.) Borrowing some notation and terminology from graph theory, if $G|_P$ denotes the path from the root r of G to the (unique) vertex $v \in G$ labelled by P via $\mathsf{l}_{\mathsf{meta}}$, then $\mathbb{L}(P, G)$ is a *minor subpath of* $G|_P$.

Given the above abstraction, we can now define protocol executions and recast the two properties of the ledger. The following definition describes two predicates, $\mathsf{ConsFail}(r_1, r_2)$ and $\mathsf{LiveFail}(r)$ that are defined on the execution graph of any particular ledger protocol execution. Looking ahead, below we introduce an experiment called the *settlement game*, which captures the influence of the adversary on a ledger protocol execution, and therefore, the adversary's influence on the corresponding execution graph. Having such a random experiment, the above predicates define corresponding events that capture security violations. Informally, consistency violation means that for two parties P_1 and P_2, P_2's ledger state is evolving in a way which is not consistent with P_1's ledger state. Analogously, liveness violation means that within a period of u rounds, the ledger state of some P receives no honest (block-)contributions.

Definition 1. *Let Π be a distributed ledger protocol and let $(\mathbb{P}, \Sigma, \mathcal{D}, \mathcal{G}, \overset{\cdot}{\dashrightarrow}, \mathbb{L})$ be the associated abstraction of Π. For any given characteristic string $w_1, \ldots, w_L \in \mathcal{W}^L$ for rounds 1 through L, let $G_0, G_1, \ldots, G_L$ denote a corresponding valid, according to $w_1, \ldots, w_L \in \mathcal{W}^L$, sequence of execution graphs (i.e., G_0 consists of only the root r, and $G_i \overset{w_{i+1}}{\dashrightarrow} G_{i+1}$ for each $i \in \{0, \ldots, L-1\}$). We define the following "bad" events (i.e., security violations):*

1. *(Consistency violation) There are $r_1 \le r_2 \le L$ and $P_1, P_2 \in \mathbb{P}$ identified in G_{r_1} and G_{r_2}, such that $\mathbb{L}(P_1, G_{r_1})$ is not a minor subpath of $G_{r_2}|_{P_2}$. We denote this event by* $\mathsf{ConsFail}(r_1, r_2)$.
2. *(Liveness violation) For some positive integer $u < L$, there is some $r \le L - u$ and a party $P \in \mathbb{P}$ that labels a (possibly different) vertex in both G_r and G_{r+u}, such that if we remove from $\mathbb{L}(P, G_{r+u})$ all vertices in $\mathbb{L}(P, G_r)$ then the remainder graph has no vertex labeled by* h *(through* $\mathsf{l}_{\mathsf{meta}}$*). We denote this event by* $\mathsf{LiveFail}_u(r)$.

Experiment $\mathbb{E}_{L,\Delta}(\mathcal{A})$.

The game is played by an adversary $\mathcal{A}^a$ and is parameterized by a duration $L \in \mathbb{N}$ and a delay Δ.

```
Let $G_0$ be the trivial execution graph containing only root.
for $i = 1$ to $L$ do
    $\mathcal{A}$ outputs $\sigma_i$ corresponding to slot $\rho_i$.
    Sample $w_i \overset{\$}{\leftarrow} \mathcal{D}(\sigma_i)$ and give $w_i$ to $\mathcal{A}$.
    $\mathcal{A}$ outputs an execution graph $G_i$ such that $G_{i-1} \xrightarrow{w_i} G_i$
end for
Set $\mathsf{ConsFail}(r_1, r_2)$ or $\mathsf{LiveFail}_u(r)$ to true accordingly if there is a consistency or liveness violation in
    $G_1, \ldots, G_L$ for some $r_1, r_2, r$.
```

[a] We use $\mathcal{A}$ for the combination of the adversary and environment.

Fig. 1. The structure of the settlement game $\mathbb{E}_{L,\Delta}(\mathcal{A})$.

The settlement game. We are now ready to reap the benefits of our formalism; we define security for a ledger protocol in a straightforward way via the settlement game in Figure 1. which captures settings where protocol participation is adversarially scheduled. The game allows the adversary to define the participation level σ of any round. [9] The participation level of the round defines the distribution of the characteristic string, e.g., lottery winner(s); the string is sampled by the challenger and the outcome is revealed to the adversary who then locally emulates his attack conditioned on that lottery outcome, and computes the execution graph G induced by this attack. The adversary wins if his attack induces a graph which indicates a consistency or liveness violation.

Assumptions underlying security. Independently of the protocol, it is impossible to expect that we can prevent consistency or liveness violations unless we restrict somehow how the adversary influences the execution. For example, it is known that Bitcoin is only secure if the majority of the hashing power invested (i.e., participating) in any given round is honest (i.e., is invested by parties following the protocol). How can we capture such conditions (and the violation thereof which is needed for defining spikes) in the above experiment?

The answer is that we will impose a restriction in Step 3 of the experiment, w.r.t. the participation symbols selected by $\mathcal{A}$. For simplicity we will consider the case where each party holds exactly one unit of resource (work or stake)—this is often referred to as the *flat model* [19]. Taking advantage of this, we can simplify Σ to contain pairs of the form (n_h, n_a). For any j, we will denote by n_a^j, to be the number of corrupted/adversarial parties and n_h^j, the number of honest parties that are up and running in round ρ_j and are synchronized with the protocol.

The flat-model assumption makes it straightforward to define the "honest (super-)majority of computing power" assumption, which underlies the security of all ledger consensus protocols: For constants $\epsilon, \theta \in (0, 1]$, we say that an adversary $\mathcal{A}$ respects the honest majority assumption with parameters θ and ϵ in rounds $\rho_i, \ldots, \rho_j$ if and only for each round $\rho \in \{\rho_i, \ldots, \rho_j\}$:

$$n_a^j \leq (1 - \epsilon) \cdot \theta \cdot n_h^j \tag{1}$$

is satisfied. Typically, we consider an arbitrary constant $\epsilon > 0$, combined with some suitable (threshold) parameter $\theta \leq 1$ that describes the required level of super-majority.

In a similar fashion, we can also impose a minimum participation constraint by requiring that $n_h^j + n_a^j > n_0$ where n_0 is a lower bound of honest party availability at any given time.

Using the above terminology, existing security claims about ledger protocols can be restated as requiring that the probability of the events $\mathsf{ConsFail}(r_1, r_2)$ and $\mathsf{LiveFail}(r)$ is negligible (in one or more relevant parameters) for any adversary which is subject to the above honest majority and participation constraints.

We remark that incrementing participation n_h^j from one round to the next does not necessarily mean that an honest party came online exactly then, since a certain time window will be necessary for an online honest party to become synchronized with the others. Instead, it means that a party has been online for sufficient time to become synchronized with the rest of the honest parties, cf. [3,2].

Remark 1. A non-flat model of resources can be captured by having participation symbols drawn from $\Sigma \subseteq [0, 1]^{\mathbb{P}} \times [0, 1]^{\mathbb{P}}$. In such case, we interpret σ as a mapping from any honest and adversarial party to the fraction of resources controlled by that party. Honest (super-)majority assumptions stated above apply verbatim to this case as well with the following interpretation for n_h, n_a: given $(\sigma_h, \sigma_a) \in \Sigma$, we set $n_h = \sum_{P \in \mathbb{P}} \sigma_h(P)$ and $n_a = \sum_{P \in \mathbb{P}} \sigma_a(P)$. We note that in this case, n_h (resp. n_a) captures the fraction of resources controlled by the honest parties collectively (resp. the adversary).

4 Adversarial Spikes and Self-Healing

The above formulation of the honest majority assumption can be directly extended to define periods of temporary adversarial majority. Since in the above experiment, it is the adversary that chooses n_a^j vs. n_h^j,

[9] In cryptographic analyses this is often the role of the execution environment, but as we are after worst case statements, we can consider the environment adversarial.

we can capture the adversary having full flexibility on how to choose which periods/rounds yield such a majority of corrupted parties (and how large such a majority is) by means of the following mechanism: We give the adversary a budget $\mathcal{B} \in \mathbb{Z}$ of total excess in resource power which he can allocate at will to different rounds. Every time a part of the budget is used, it is removed from $\mathcal{B}$; then, while choosing n_a^j and n_h^j, the adversary is allowed to violate the honest majority condition (1) by a total of at most $\mathcal{B}$: i.e., he can propose a pair (n_h^j, n_a^j), such that $n_a^j = \lfloor (1-\epsilon) \cdot \theta \cdot n_h^j \rfloor + \mathrm{surp}_j$, as long as surp_j is less than the current value of $\mathcal{B}$. We refer to an adversary that respects this budget restriction and never drops the total participation below some n_0 as a $(\theta, \epsilon, n_0, \mathcal{B})$-*adversary.*

Definition 2 ($(\theta, \epsilon, n_0, \mathcal{B})$-adversary). *Let $\theta, \epsilon \in (0, 1]$ and $\mathcal{B} \in \mathbb{N}$. A $(\theta, \epsilon, n_0, \mathcal{B})$-adversary is an adversary in the settlement game of Figure 1 that satisfies the following properties: Let $\mathcal{B}_0 := \mathcal{B}$. In each iteration i of the for-loop, the adversary might output a pair (n_h^i, n_a^i) satisfying $n_a^i + n_h^i \geq n_0$ such that $n_a^i \leq (1-\epsilon) \cdot \theta \cdot n_h^i + surp_i$, where $0 \leq surp_i \leq \mathcal{B}_{i-1}$; furthermore, at the end of the i-th iteration, we set $\mathcal{B}_i := \mathcal{B}_{i-1} - surp_i$, to be the* residual budget *that the adversary has to use after ρ_i.*

For $\mathcal{B} > 0$, we say that a $(\theta, \epsilon, n_0, \mathcal{B})$-adversary $\mathcal{A}$ performs his attack between rounds $\rho_\alpha < \rho_\beta$, if the following conditions hold: (1) $\mathcal{B}_\ell = \mathcal{B}$ for all $\rho_\ell < \rho_\alpha$ and $\mathcal{B}_\alpha < \mathcal{B}$; (2) $\mathcal{B}_\beta > 0$, and $\mathcal{B}_{\beta+1} = 0$; we refer to round ρ_α as the first attack round*, to round ρ_β as the* last attack round*, and to round $\rho_\beta + 1$ as the* first healing (period) round*.*

Given the above definition we next define the self-healing properties of a ledger protocol with respect to consistency and liveness.

Definition 3. *(Self-Healing Ledger Protocol) Let Π be a ledger protocol and $(\mathbb{P}, \Sigma, \mathcal{D}, \mathbb{G}, \dot{\rightarrow}, \mathbb{L})$ be the associated abstraction. We say that Π is* self-healing with vulnerability period defined by the pair τ_l and τ_h *w.r.t. consistency (resp. liveness) against a $(\theta, \epsilon, n_0, \mathcal{B})$-adversary who performs his attack between rounds ρ_α and ρ_β, if the event* $\mathsf{ConsFail}(r, \cdot)$ *(resp.* $\mathsf{LiveFail}(r)$*) in Figure 1 occurs with at most negligible probability, unless $r \in \{\rho_\alpha - \tau_\mathrm{l}, \ldots, \rho_\beta + \tau_\mathrm{h}\}$.*

The vulnerability period is typically a function of the budget $\mathcal{B}$, the duration of the spike itself, and other parameters including the nominal stake distribution. The self-healing definition is a direct relaxation of the security definition of a ledger protocol (in the sense that it allows the events $\mathsf{ConsFail}(r_1, r_2)$ (resp. $\mathsf{LiveFail}(r)$) to happen at some point of the execution which is related to the adversarial majority attack. Importantly, τ_l and τ_h yield intuitive bounds on how much history the considered potential adversarial spike can overwrite—where history corresponds to rounds before the spike starts—and how much time the protocol needs before its consistency and liveness are properly restored after a spike(s)-inducing attack ends. Note that liveness and consistency behave slightly differently at the onset of a spike: while the spike can "retroactively" interfere with consistency, liveness is ensured until the spike commences.

Notational conventions. Throughout the paper, let $\mathbb{N} = \{0, 1, 2, \ldots\}$ denote the set of natural numbers (including zero). For $n \in \mathbb{N}$, $[n]$ denotes the set $\{1, \ldots, n\}$ (hence $[0] = \emptyset$). For a word w over some alphabet Σ, i.e, $w = w_1 \ldots w_n \in \Sigma^n$, we denote by $w_{i:j}$ its subword $w_i w_{i+1} \ldots w_j$, and by $\#_a(w)$ we denote the number of ocurrences of the symbol $a \in \Sigma$ in w. We extend this notation also to random variables.

5 Nakamoto-Style Proof of Work

5.1 Preliminary Considerations and Notation

In Nakamoto-style PoW ledger protocols, the participants (aka miners) keep evaluating a hash function on appropriate different inputs until someone wins (i.e., finds a value of the function below a predefined target; an event of probability p). We provide next the full specification of Nakamoto-style PoW protocols within our ledger abstraction $(\mathbb{P}_{\mathsf{pow}}, \Sigma_{\mathsf{pow}}, \mathcal{W}_{\mathsf{pow}}, \mathcal{D}_{\mathsf{pow}}, \mathcal{G}_{\mathsf{pow}}, \dot{\rightarrow}_{\mathsf{pow}}, \mathbb{L}_{\mathsf{pow}})$. All of these elements, except the execution graph and its transition relation, are rather clear: (i) the set $\mathbb{P}_{\mathsf{pow}}$ is an arbitrary name space to distinguish protocol participants. Therefore, $n := |\mathbb{P}_{\mathsf{pow}}|$ denotes the upper bound on the number of parties.

(ii) $\varSigma_{\mathsf{pow}} = \mathbb{N} \times \mathbb{N}$ are the numbers to denote the active honest and adversarial hashing units with the restriction that they sum up to a number less or equal to n. (iii) $\mathcal{W}_{\mathsf{pow}} = \mathbb{N} \times \mathbb{N}$ denotes the number of hash-puzzle successes for honest and adversarial entities. Hence, (iv) $\mathcal{D}_{\mathsf{pow}}$, following a flat model of participation, samples (h, a) as the number of Bernoulli successes in $(\sigma_1, \sigma_2) \in \varSigma$ trials respectively (see Definition 8 for the actual distribution). Therefore, the *characteristic string* w of an execution is drawn from the set $(\mathcal{W}_{\mathsf{pow}})^L$ and we can write

$$w = (w_1, \ldots, w_L) = ((h_1, a_1), \ldots, (h_L, a_L))$$

so that the t-th symbol of w is (h_t, a_t) and indicates a round during which there were h_t honest PoW successes and a_t adversarial PoW successes. Finally, (v) $\mathbb{L}_{\mathsf{pow}}(G, P)$ outputs the sub-path of $G|_P$ that drops the last k block production events, capturing the fact that parties in Nakamoto-style PoW blockchain protocols chop off a number of blocks in the suffix of the blockchain they possess to arrive at a common prefix. For simplicity we adopt κ as a single parameter to be used when determining the length of this suffix, cf. [19].

The most involved objects in the above characterization are the execution graph and the transition relation between execution graphs from one round to the next, these are discussed next.

5.2 PoW Execution Graphs and the Fork Abstraction of Nakamoto-Style Blockchains

We now define a combinatorial structure that allows us to analyze the settlement game. This structure is a generalization of the "fork" concept considered in the literature [23,11,2,17].

PoW execution graph. We next define the remaining two objects $\mathcal{G}_{\mathsf{pow}}, \dot{\to}_{\mathsf{pow}}$ in the ledger-abstraction tuple, which fully defines the settlement game for Nakamoto-style PoW blockchains. The execution graph $\mathcal{G}_{\mathsf{pow}}$ for each round tracks block production events connecting them with an edge when a block extends a previous block. For this, it is sufficient to have the metadata labeling $\mathsf{l}_{\mathsf{meta}}$ over $\mathcal{W}_{\mathsf{pow}} \times \{\mathsf{h}, \mathsf{a}\} \times 2^{\mathbb{P}}$ and leave AUX empty: the relevant information is whether an honest participant (h) or the adversary (a) produced a block. Complying with the model, each vertex must be labeled by the symbol of the characteristic string of the round it was appended, and possibly a set of party identifiers, which in the PoW case denotes the set of parties that adopt the chain from the root up to that node in the tree.

The relation $\overset{w_i}{\dot{\to}}_{\mathsf{pow}}$ subjects the transition from an execution graph G_{i-1} to G_i to the following constraints: (a) $w_i = (h_i, a_i)$ specifies the upper bound a_i of new nodes in G_i that are labeled with a and the exact amount h_i of new nodes labeled with h. (b) whenever a party $P \in \mathbb{P}_{\mathsf{pow}}$ appears in G_i it either appears also in G_{i+1} or it never appears in any G_j for $j > i$, (c) whenever a party $P \in \mathbb{P}_{\mathsf{pow}}$ appears in G_i, G_{i+1} the assignment of P in G_{i+1} should be at either the same or a longer path compared to G_i, (d) whenever a party $P \in \mathbb{P}_{\mathsf{pow}}$ appears in G_i the assignment of P is at least as long as the longest path ending in an honest node labeled h, $\varDelta$ rounds ago.

Simplification: the fork abstraction. While the full specification of execution-graphs given above is necessary to be able to define the self-healing properties we are after, it will be most convenient in the analysis to handle a slightly simpler object. A "fork" is a condensed form of the execution graph which drops the $\mathcal{W}_{\mathsf{pow}}$ and $2^{\mathbb{P}}$ annotations leaving just the type of a node, either honest or adversarial — we will denote this stripped down version of $\mathsf{l}_{\mathsf{meta}}$ by $\mathsf{l}_{\mathsf{type}}$. As for the missing information, the characteristic string symbol w_i is usually clear from the context, while the party identifiers assigned to a node can be ignored by focusing only at any given time on so-called *dominant paths* which roughly speaking are all those paths that end in an honest node labeled h $\varDelta$ rounds ago (the reason is that all honest parties must be assigned to such nodes by the definition of $\dot{\to}_{\mathsf{pow}}$). Based on this, we define the set of valid sequences of forks appearing in the settlement game for a PoW-blockchain protocol as follows:

Definition 4 (PoW $\varDelta$-fork). *Let $\varDelta$ be a positive integer and $L \in \mathbb{N}$. A* PoW $\varDelta$-fork *for the string $w \in \mathcal{W}_{\mathsf{pow}}^L$ is a directed, rooted tree $F = (V, E)$ with a pair of functions*

$$\mathsf{l}_{\#} : V \to \mathbb{N} \qquad \textit{and} \qquad \mathsf{l}_{\mathsf{type}} : V \to \{\mathsf{h}, \mathsf{a}\}$$

satisfying the axioms below. Edges are directed "away from" the root so that there is a unique directed path from the root to any vertex. The value $\mathsf{l}_{\#}(v)$ is referred to as the label *of v. The value $\mathsf{l}_{\mathsf{type}}(v)$ is referred to as the* type *of the vertex: when $\mathsf{l}_{\mathsf{type}}(v) = \mathsf{h}$, we say that the vertex is* honest*; otherwise it is* adversarial*.*

(i) the root $r \in V$ is honest and has label $\mathsf{l}_{\#}(r) = 0$;

(ii) the sequence of labels $\mathsf{l}_{\#}()$ along any directed path is non-decreasing; the sequence of labels $\mathsf{l}_{\#}()$ appearing on the honest vertices of any directed path is strictly increasing;

(iii) if $w_i = (h_i, a_i)$, there are exactly h_i honest vertices with the label i and no more than a_i adversarial vertices of F with the label i;

(iv) for any pair of honest vertices v, w for which $\mathsf{l}_{\#}(v) + \Delta \leq \mathsf{l}_{\#}(w)$, $\mathsf{len}(v) < \mathsf{len}(w)$, where $\mathsf{len}()$ denotes the depth of the vertex.

We note that the notion is very close to the notion of PoS-forks introduced in [11] which we use and recall in Section 6.2. When no confusion can arise, we will refer to both of these objects simply as "forks" (treating also the parameter Δ implicitly). We now recall several definitions involving forks that apply to both, PoW and PoS forks, including the following:

Definition 5 (Fork notation). *We write $F \vdash_{\Delta} w$ to indicate that F is a Δ-fork for the string w. When $\Delta = 1$, corresponding to the synchronous case, we may just write $F \vdash w$. If $F' \vdash_{\Delta} w'$ for a prefix w' of w, we say that F' is a* subfork *of F, denoted $F' \sqsubseteq F$, if F contains F' as a consistently-labeled subgraph. A fork $F \vdash_{\Delta} w$ is* closed *if all leaves are honest. By convention the trivial fork, consisting solely of a root vertex, is closed. The* closure *of a fork F, denoted $\overline{F}$, is the maximal closed subfork of F.*

Definition 6 (Tines). *A path in a fork F originating at the root is called a* tine *(note that tines do not necessarily terminate at a leaf). As there is a one-to-one correspondence between directed paths from the root and vertices of a fork, we routinely overload notation so that it applies to both tines and vertices. Specifically, we let $\mathsf{len}(T)$ denote the* length *of the tine, equal to the number of edges on the path; we additionally use this same notation $\mathsf{len}(v)$ to indicate the depth of a vertex. In the unusual cases where we wish to emphasize the fork from which v is drawn, we write $\mathsf{len}_F(v)$. We further overload this notation by letting $\mathsf{len}(F)$ denote the length of the longest tine in a fork F. Likewise, we let $\mathsf{l}_{\#}(\cdot)$ apply to tines by defining $\mathsf{l}_{\#}(T) \triangleq \mathsf{l}_{\#}(v)$, where v is the terminal vertex on the tine T.*

For a vertex v in a fork F, we denote by $F(v)$ the tine in F terminating in v. For two tines T, T' of a fork F, we write $T \sim_{\ell} T'$ if the two tines share a vertex with a label greater or equal to ℓ. For a vertex v, we call the length of the tine terminating at v the depth *of v. We say that a tine is* honest *if the last vertex of the tine is labeled with a uniquely honest index.*

Definition 7 (Fork trimming; dominance). *For a string $w = w_1 \ldots w_n$ and a positive integer k, we let $w_{\lceil k} = w_1 \ldots w_{n-k+1}$ denote the string obtained by removing the last $k-1$ symbols. For a fork $F \vdash_{\Delta} w_1 \ldots w_n$ we let $F_{\lceil k} \vdash_{\Delta} w_{\lceil k}$ denote the fork obtained by retaining only those vertices labeled from the set $\{1, \ldots, n-k+1\}$. Observe that honest tines appearing in $F_{\lceil \Delta}$ are those that are necessarily visible to honest players at a round just beyond the last one described by the characteristic string. Similarly, we say that a tine T in F is Δ-*dominant *if $\mathsf{len}(T) \geq \mathsf{len}(\overline{F_{\lceil \Delta}})$ and simply call it* dominant *if Δ is clear from the context.*

Intuitively, the distribution $\mathcal{D}_p(n_h, n_a)$ defined below gives the probability of a particular outcome in a round with n_h honest and n_a adversarial parties, where the success probability of a single mining query is p.

Definition 8. *Fix $p \in (0,1)$ and $n_h, n_a \in \mathbb{N}$. We denote by $\mathcal{D}_p(n_h, n_a)$ the probability distribution over $\mathcal{W}_{\mathsf{pow}}$ defined by*

$$\Pr_{\mathcal{D}_p(n_h,n_a)}[(h,a)] \triangleq \binom{n_h}{h}\binom{n_a}{a} p^{h+a}(1-p)^{n_h+n_a-h-a} . \tag{2}$$

5.3 Asymptotic Analysis

Advantage and margin. We develop some tools for reasoning about the settlement game. For a Δ-fork $F \vdash_\Delta w$, we define the Δ-*advantage* of a tine $T \in F$ as

$$\alpha_F^\Delta(T) = \mathsf{len}(T) - \mathsf{len}(\overline{F}_{\lceil \Delta}) \,. \tag{3}$$

Observe that $\alpha_F^\Delta(T) \geq 0$ if and only if T is Δ-dominant in F. For $\ell \geq 1$, we define the quantity of interest

$$\beta_\ell^\Delta(F) = \max_{\substack{T \not\sim_\ell T^* \\ T^* \text{ is } \Delta\text{-dominant}}} \alpha_F^\Delta(T) \,,$$

this maximum extended over all pairs of tines (T, T^*) where T^* is Δ-dominant and $T \not\sim_\ell T^*$. Note that there might exist multiple such pairs in F, but under the condition $\ell \geq 1$ there will always exist at least one such pair, as the trivial tine T_0 containing only the root vertex satisfies $T_0 \not\sim_\ell T$ for any T and $\ell \geq 1$, in particular $T_0 \not\sim_\ell T_0$. For this reason, we will always consider β_ℓ^Δ only for $\ell \geq 1$.

We overload the notation and let

$$\beta_\ell^\Delta(w) = \max_{F \vdash_\Delta w} \beta_\ell^\Delta(F) \,.$$

Intuitively, $\alpha_F^\Delta(T)$ captures the length advantage (or deficit) of the tine T against the longest honest tine created at least Δ slots before the upcoming slot, and hence now known to all honest parties. Consequently, $\beta_\ell^\Delta(F)$ records the maximal advantage of any tine T_a in F that potentially disagrees with some Δ-dominant tine T_h about the chain state up to slot ℓ.

The crucial property motivating the above definition is that if an execution of a PoW blockchain up to round $t > \ell$ has resulted into a fork F, as all honest parties at time t are holding a chain of length at least $\mathsf{len}(\overline{F}_{\lceil \Delta})$, a negative value od $\beta_\ell^\Delta(F) < 0$ then indicates that the adversary cannot make them switch to any chain of at least the same length that would disagree with their chain before slot ℓ. If $\beta_\ell^\Delta(w) < 0$ then no fork allowing for such a consistency failure (with respect to the state up to slot ℓ) even exists for the string w. This connection between $\beta_\ell^\Delta(w)$ and our predicate $\mathsf{ConsFail}$ is finally employed in the proof of Theorem 1 and is established along the lines of the reasoning used in [17] (where a slightly different notion of consistency is used).

An exact analysis in the serialized, synchronous setting. We begin with an analysis of the quantity $\beta_\ell^\Delta(w)$ in a simple synchronous setting, corresponding to the case when $\Delta = 1$ and block creation is strictly serialized. Specifically, we work with a reduced alphabet $\mathcal{W}'_{\mathsf{pow}} = \{(1,0),(0,1)\}$ for characteristic strings, and use the abbreviations $\mathsf{h} = (1,0)$ and $\mathsf{a} = (0,1)$; thus we treat characteristic strings over the alphabet $\{\mathsf{h}, \mathsf{a}\}$. The definition of fork is unchanged. In this synchronous case we abbreviate $\alpha_F^1()$ by $\alpha_F()$, and note that $\alpha_F(T) = \mathsf{len}(T) - \mathsf{len}(\overline{F})$. We similarly abbreviate β_ℓ^1 by $\beta_\ell()$.

Lemma 1. *Fix $\ell \geq 1$. We consider characteristic strings $w \in \{\mathsf{h}, \mathsf{a}\}^*$. By definition $\beta_\ell(\varepsilon) = 0$. In general,*

$$\beta_\ell(w\mathsf{a}) = \beta_\ell(w) + 1 \,, \quad \textit{and} \quad \beta_\ell(w\mathsf{h}) = \begin{cases} \beta_\ell(w), & \textit{if } \beta_\ell(w) = 0 \textit{ and } |w\mathsf{h}| < \ell, \\ \beta_\ell(w) - 1, & \textit{otherwise} \end{cases} \tag{4}$$

Thus, prior to round ℓ this is a biased *barrier walk* with a barrier at 0 (cf. Def. 12 below). In contrast, *after* round ℓ, this just behaves like a conventional biased random walk.

Proof. We proceed by induction on the length of w. The base case is immediate, as the trivial fork has no nontrivial tines.

We begin by establishing the lower bounds for the quantities $\beta_\ell(w\mathsf{a})$ and $\beta_\ell(w\mathsf{h})$ corresponding to each of the equations (4) above. These implicitly yield an optimal (on-line) adversary for maximizing $\beta_\ell()$: specifically, for a characteristic string $w_1, \ldots, w_n$, this yields a sequence of forks $F_1 \sqsubseteq \cdots \sqsubseteq F_n$ so that $F_t \vdash w_1 \ldots w_t$ and each F_t is only determined by the string $w_1 \ldots w_t$. We then turn to the corresponding upper bounds, which establish equality in each of the cases above.

Bounding from below; the optimal adversary. Let w be a characteristic string and $F \vdash w$ a fork achieving $\beta_\ell(F) = \beta_\ell(w)$; let T and T^* be two tines of F which witness β so that T^* is dominant and $\alpha_F(T) = \beta_\ell(F)$. Then:

1. Consider the fork $F' \vdash w\mathsf{a}$ obtained by extending the tine T with a new adversarial vertex labeled with the last symbol. This new fork achieves $\beta_\ell(F') \geq \beta_\ell(w) + 1$.
2. If $\beta_\ell(w) \neq 0$, consider the fork $F' \vdash w\mathsf{h}$ obtained by adding an honest vertex to the unique vertex on T^* of depth $\mathsf{len}(\overline{F})$. This new fork achieves $\beta_\ell(F') \geq \beta_\ell(w) - 1$. (A pair of tines that witness this in F' are T and the dominant tine terminating in the new vertex.)
3. If $\beta_\ell(w) = 0$ and $|w\mathsf{h}| < \ell$, consider an arbitrary fork $F' \vdash w\mathsf{h}$ for which $F \sqsubseteq F'$. As $|w\mathsf{h}| < \ell$ any dominant tine in F' can serve as both T^* and T in the definition of β_ℓ. This achieves $\beta_\ell(F') \geq 0$.
4. If $\beta_\ell(w) = 0$ and $|w\mathsf{h}| \geq \ell$, this proceeds as in case (2) above.

Bounding from above. To complete the proof, we establish the opposite inequalities in each of these cases.

The cases for $w\mathsf{h}$. Let $F' \vdash w\mathsf{h}$ be a fork for which $\beta_\ell(F') = \beta_\ell(w\mathsf{h})$; let T and T^* be tines that witness this value of β, where T^* is dominant and $\alpha_{F'}(T) = \beta_\ell(F')$. Let $F \vdash w$ be the fork obtained by removing the honest vertex v of F' associated with the final h symbol. Observe that $\mathsf{len}(\overline{F}) \leq \mathsf{len}(\overline{F'}) - 1$.

If v does not appear on T, this tine remains in F and $\alpha_F(T) \geq \beta_\ell(w\mathsf{h}) + 1$. The tine T^* might not appear in F (if v appears on T^*). In any case, however, the restriction of T^* to the fork F always has length at least $\mathsf{len}(\overline{F})$ and hence is dominant. We conclude that $\beta_\ell(w) \geq \beta_\ell(w\mathsf{h}) + 1$.

Otherwise v appears on T, in which case T is an honest tine and $\beta_\ell(w\mathsf{h}) = \alpha_{F'}(T) = 0$. If the tines T and T^* are distinct, we may switch their roles (as both are dominant) and apply the argument above to conclude that $\beta_\ell(w) \geq \beta(w\mathsf{h}) + 1$. Otherwise $T = T^*$ and we conclude that $|w\mathsf{h}| < \ell$. In this case, removing the last vertex from these tines results in a tine $\check{T}$ for which $\alpha_F(\check{T}) = 0$, establishing $\beta_\ell(w) \geq 0$. Hence, considering separately the cases $\beta_\ell(w) > 0$ and $\beta_\ell(w) = 0$, in each of them the desired inequality holds.

The case $w\mathsf{a}$. Let $F' \vdash w\mathsf{a}$ realize $\beta_\ell(F') = \beta_\ell(w\mathsf{a})$ and let T and T^* be two tines of F' that witness $\beta_\ell(F')$, as above.

We begin with an argument showing that the fork F' can be restructured to yield a fork $\widehat{F} \vdash w\mathsf{a}$ for which $\beta_\ell(\widehat{F}) = \beta_\ell(F')$ and there is a pair of tines $\widehat{T}$ and $\widehat{T}^*$ witnessing this value of $\beta_\ell(\widehat{F})$ with the property that $\widehat{T}$ terminates with the adversarial vertex v associated with the last symbol a, $\alpha_{\widehat{F}}(\widehat{T}) = \beta_\ell(w\mathsf{a})$, and $\widehat{T}^*$ is dominant.

Restructuring F'. If T contains the vertex v, F' already has the desired property. Otherwise the vertex v must, in fact, appear on T^*: if it appeared on neither T nor T^*, removing the vertex from the end of the tine on which it appears (if it exists) and adding it to the end of the tine T would result in a larger $\beta_\ell()$. To construct the fork $\widehat{F}$, let v_h denote the honest vertex of maximum depth among those vertices on either T or T^*. Let $T_h \in \{T, T^*\}$ be a tine containing v_h, and T_a denote the other tine. $\widehat{F}$ is constructed from F' as follows: (I.) All adversarial vertices on T_h appearing after v_h are removed from T_h and inserted into the tine T_a, producing a new tine $\widehat{T}$; this is possible because T_a has no honest vertex with label larger than $\mathsf{l}_\#(v_h)$. Observe that the tine $\widehat{T}$ constructed in this way contains the final adversarial vertex v. (II.) Starting with the vertex v_h, construct a tine $\widehat{T}^*$ by appending to the path all honest vertices appearing after v_h, in order. Any other (necessarily adversarial) vertices orphaned by this process can be attached to the fork arbitrarily. This constructs a new fork $\widehat{F} \vdash w\mathsf{a}$.

As $T \not\sim_\ell T^*$, it is clear that the tines $\widehat{T}$ and $\widehat{T}^*$ constructed above inherit this property. Note, also, that $\widehat{T}^*$ is clearly dominant in $\widehat{F}$, as it terminates with the deepest honest vertex of $\widehat{F}$. It remains to ensure that $\alpha_{\widehat{F}}(\widehat{T}) \geq \beta_\ell(F')$. Recall that $T_a \in \{T, T^*\}$. If $T_a = T$, it is clear that $\alpha_{\widehat{F}}(\widehat{T}) \geq \alpha_{F'}(T)$ because $\mathsf{len}(\overline{\widehat{F}}) \leq \mathsf{len}(\overline{F'})$ and $\mathsf{len}(\widehat{T}) \geq \mathsf{len}(T)$ by construction. In the other case $T_a = T^*$, any adversarial vertices were inserted into the tine T^* (to yield $\widehat{T}$). Recall that T^* was dominant in F'; if $\beta_\ell(w\mathsf{a}) \leq 0$, this immediately yields $\alpha_{\widehat{F}}(T^*) \geq \beta_\ell(w\mathsf{a})$, as desired. Otherwise, observe that the number of adversarial vertices inserted in T^* is at least $\alpha_{F'}(T) = \beta_\ell(w\mathsf{a})$, in which case it is clear that $\alpha_{\widehat{F}}(\widehat{T}) \geq \beta_\ell(F')$, as desired. This completes the construction and its analysis.

To complete the argument, assume that the fork F' possesses the property guaranteed above (that the final adversarial vertex appears on the tine T). Let $F \sqsubseteq F'$ denote the fork $F \vdash w$ obtained by removing the adversarial vertex v associated with the final symbol a. Then the restriction of T to F and the tine T^* together witness the fact that $\beta_\ell(w) \geq \beta_\ell(F) \geq \beta_\ell(F') - 1 = \beta_\ell(w\mathsf{a}) - 1$, as desired. □

The serialization and reduction mapping. We relate $\beta_\ell()$ to $\beta_\ell^\Delta()$ via a *reduction and serialization* relation, a technique developed in David et al. [11]. This permits an analysis of the Δ-semi-synchronous setting via a reduction to the synchronous setting. Consider a characteristic string $w = ((h_1, a_1), \ldots, (h_L, a_L)) \in (\mathcal{W}_{\mathsf{pow}})^L$: an index i is *isolated* if there is no more than one honest PoW success among rounds j for which $|i - j| \leq \Delta$, where both 0 and $L+1$ are interpreted as rounds with honest PoW successes; formally, $i > 0$ is isolated if $\Delta \leq i \leq L - \Delta + 1$ and $\sum_{j, |i-j| \leq \Delta} h_j \leq 1$, where h_0 and h_{L+1} are treated as 1 for the purposes of the sum. Otherwise the index is said to be *crowded*.

For a fixed value of Δ, we introduce a relation $\sim$ over $\mathcal{W}_{\mathsf{pow}}^*$ and $\{\mathsf{h}, \mathsf{a}\}^*$ given by the following rule. Let $w \in \mathcal{W}_{\mathsf{pow}}^L$; for a symbol $w_i = (h_i, a_i)$ of w, define the subset $\mathcal{X}_i \subseteq \{\mathsf{h}, \mathsf{a}\}^*$ so that

$$\mathcal{X}_i = \begin{cases} \{x \in \{\mathsf{h}, \mathsf{a}\}^{a_i + 1} \mid \#_{\mathsf{h}}(x) = 1\} & \text{if } i \text{ isolated, } h_i = 1; \\ \{\mathsf{a}^{a_i}\} & \text{otherwise.} \end{cases} \tag{5}$$

Then we define $w \sim x$ if and only if $x \in \mathcal{X}_1 \circ \cdots \circ \mathcal{X}_L$, where $\circ$ denotes concatenation of languages. It is notationally more convenient to treat $\sim$ as a function: we write $\rho^\Delta(w) = \{x \mid w \sim x\} \subseteq \{\mathsf{a}, \mathsf{h}\}^*$. As a fixed round ℓ plays a distinguished role in our analysis, we slightly adapt this relation to separate the portions of the reduced characteristic string that arise from $w_1 \ldots w_\ell$ and $w_{\ell+1} \ldots$. Specifically, we define

$$\rho_\ell^\Delta(w) = \left\{ (x, y) \;\middle|\; \begin{array}{l} x \in \mathcal{X}_1 \circ \cdots \circ \mathcal{X}_\ell; \\ y \in \mathcal{X}_{\ell+1} \circ \cdots \circ \mathcal{X}_L \end{array} \right\}.$$

In preparation for the next lemma, we also define the Δ-residue of a characteristic string w, equal to the number of honest PoW successes in the last $\Delta - 1$ rounds:

$$\mathrm{res}_\Delta(w) = \sum_{i > |w| - \Delta} h_i \,.$$

Lemma 2. *Let $w = ((h_1, a_1), \ldots, (h_L, a_L)) \in \mathcal{W}_{\mathsf{pow}}^L$. Then*

$$\beta_\ell^\Delta(w) \leq \max_{(x,y) \in \rho_\ell^\Delta(w)} \beta_{|x|}(xy) + \mathrm{res}_\Delta(w) \,.$$

Proof. Let $F \vdash_\Delta w$; the first step of the proof constructs a related (synchronous) fork $F' \vdash_1 w'$ for a related string w' obtained from w by removing honest vertices with crowded labels, with the guarantee that $\beta_\ell^\Delta(F) \leq \beta_\ell(F') + \mathrm{res}_\Delta(w)$. The second step of the proof constructs from F' a fork $F'' \vdash_1 w''$ that is fully serialized in the sense that $w'' \in \{\mathsf{h}, \mathsf{a}\}^*$.

Step I: The reduction to synchrony. Let

$$D = \{\mathsf{len}(v) \mid v \text{ honest, } \mathsf{l}_\#(v) \text{ is crowded}\} \,,$$

denote the set of depths of those honest vertices appearing at crowded rounds. Let F' denote the graph obtained from F by contracting all edges terminating at vertices with depths in the set D. Recall that contracting the directed edge (u, v) replaces the two vertices u and v with a single vertex u' which inherits all edges incident upon u or v. (The edge (u, v) is removed.) Globally, this has the effect of removing all vertices, along with their incident edges, appearing at depths in the set D, and appropriately introducing edges corresponding to all paths that were thus removed. As the root vertex is never considered crowded, it is easy to see that the resulting graph has the structure of a directed tree. The effect of this process on relative depths will be relevant for the remainder of the argument, so we briefly record a few immediate properties: If u and v are vertices of F that survive in F',

1. $|\mathsf{len}_{F'}(u) - \mathsf{len}_{F'}(v)| \leq |\mathsf{len}_F(u) - \mathsf{len}_F(v)|$,
2. $\mathsf{len}_F(u) = \mathsf{len}_F(v) \Rightarrow \mathsf{len}_{F'}(u) = \mathsf{len}_{F'}(v)$, and
3. $\mathsf{len}_F(u) > \mathsf{len}_F(v) \Rightarrow \mathsf{len}_{F'}(u) > \mathsf{len}_{F'}(v)$.

The resulting fork F' is not, in general, a legal Δ-fork for the string w; however, it is a synchronous fork (that is, a 1-fork) for the related string $w' = ((h'_1, a'_1), \ldots, (h'_L, h'_L))$ defined so that

$$a'_i = a_i \quad \text{and} \quad h'_i = \begin{cases} h_i & \text{if } i \text{ is isolated,} \\ 0 & \text{if } i \text{ is crowded.} \end{cases}$$

Here F' directly inherits the labelings from F. It remains to check that honest vertices in F' satisfy the synchronous monotonicity condition. Note, however, that any surviving honest vertex v was isolated in F and hence has depth exceeding that of any prior honest vertex. It follows that $F' \vdash w'$ (with $\Delta = 1$).

To evaluate the effect this transformation has on $\beta_\ell()$, consider two tines T and T^* that witness $\beta_\ell^\Delta(F)$; as usual, we assume T^* is Δ-dominant and $\alpha_F^\Delta(T) = \beta_\ell^\Delta(F)$. These two tines have direct analogues, T' and T'^* in F', obtained by "jumping over" all vertices in F with depths in D.

Observe that T'^* is 1-dominant in F'; here we critically use the fact that any honest vertex associated with the last $\Delta - 1$ rounds of w will necessarily have been treated as crowded and so removed from F. Thus, $\mathsf{len}_F(T^*) \geq \mathsf{len}_F(v)$ for any isolated honest vertex v of F; it follows that $\mathsf{len}_{F'}(T'^*) \geq \mathsf{len}_{F'}(v)$. Note, furthermore, that $T'^* \not\sim_\ell T'$ so that $\alpha_{F'}(T') \leq \beta_\ell(F')$.

It remains to control $\alpha_{F'}(T')$. In case $\mathsf{len}_F(T) \leq \mathsf{len}_F(T^*)$ we find that $\beta_\ell^\Delta(F) \leq \beta_\ell(F')$, as desired. Otherwise, $\mathsf{len}_F(T) > \mathsf{len}_F(T^*)$ and we may incur a penalty in the relative lengths equal to the number of depths appearing in T beyond $\mathsf{len}_F(T^*)$ that appear in D. As T^* was Δ-dominant in F, this can be no more than $\mathrm{res}_\Delta(w)$. We conclude that $\beta_\ell(F') \geq \beta_\ell^\Delta(F) - \mathrm{res}_\Delta(w)$.

Step II: Serialization. We reconsider the synchronous fork

$$F' \vdash_1 w' = ((h'_1, a'_1), \ldots, (h'_L, a'_L))$$

defined above. To complete the proof, we may naturally treat F' as a fork for a related characteristic string $w'' \in \{(1,0),(0,1)\}^* = \{\mathsf{h}, \mathsf{a}\}^*$ by appropriately serializing the vertices associated with each symbol (h'_i, a'_i) of w'.

More precisely, define $M_k = \sum_{i=1}^k h'_i + a'_i$ and $M = M_L$. Let $\mathsf{l}_\#$ denote the labeling function of $F' = (V, E)$, we construct an alternate labeling $\mathsf{l}_\#^S : V \to \{1, \ldots, M\} = [M]$ for F' that effectively serializes the fork in the sense that (i.) the labeling is injective: each $i \in [M]$ labels no more than one vertex, and (ii.) $\mathsf{l}_\#^S$ is strictly increasing along tines in F'. The serializing labeling also maintains the invariant that any vertices for which $\mathsf{l}_\#(v) = k$ are assigned (injectively) to the ($\mathsf{l}_\#^S$) labels in the set $\{M_{k-1}+1, \ldots, M_k\}$. The existence of such a labeling follows directly by induction: Note that F' induces a partial order on V by the rule $u \leq v \Leftrightarrow \exists$ a directed path from u to v; the desired labeling is given by refining the partial order induced by F' on vertices for which $\mathsf{l}_\#(v) = k$ to a total order. (Note that there may be circumstances where F' has fewer than $h'_i + a'_i$ vertices associated with $\mathsf{l}_\#(v) = i$). We denote by F'' the fork F' with the labeling $\mathsf{l}_\#$ replaced by $\mathsf{l}_\#^S$. We then construct the characteristic string $w'' = (w''_1, \ldots, w''_M)$ so that

$$w''_i = \begin{cases} \mathsf{h} & \text{if } \mathsf{l}_\#^S(v) = i \text{ for an honest } v, \\ \mathsf{a} & \text{otherwise.} \end{cases}$$

(Recall that if there is a vertex v for which $\mathsf{l}_\#^S(v) = i$, then there is a unique such vertex.) Clearly $F'' \vdash w''$. Recalling the definition of the sets $\mathcal{X}_i$ as given in (5) above, we see that $w'' \in \mathcal{X}_1 \circ \cdots \circ \mathcal{X}_L$. By separating the portion of the string w'' that is derived based on $w_1 \ldots, w_\ell$ (as done above) from the suffix, and denoting the resulting split as $w'' = xy$, we obtain $(x, y) \in \rho_\ell^\Delta(w)$. As the graph structure of the fork F'' is identical to F', we have $\beta_\ell(F') \leq \beta_{|x|}(xy)$, concluding the proof of the lemma. □

Probabilistic preliminaries. Before proceeding to the stochastic analysis of the relevant random walks, we introduce a few useful definitions and facts.

Definition 9. *Let Ω be a partial order under the relation $\leq$. A subset $S \subset \Omega$ is* monotone *if, for all $x, y \in \Omega$, $x \in S \wedge x \leq y \Rightarrow y \in S$. Let X and Y be two random variables taking values in Ω. We say that Y* stochastically dominates *X, written $X \prec Y$, if $\Pr[Y \in S] \geq \Pr[X \in S]$ for all monotone sets S.*

In our setting, the primary use of stochastic dominance is to prove tail bounds—of the form $\Pr[X \geq k]$ where X takes real values—by studying a simpler random variable Y and using the fact that $\Pr[X \geq k] \leq \Pr[Y \geq k]$. We formulate this so generally because there are circumstances where we need dominance for random variables defined over other partial orders.

Definition 10. *Let $n \in \mathbb{N}$ and $p \in [0, 1]$. A random variable $X \in \{0, \ldots, n\}$ has the* binomial distribution *$\mathcal{B}_{n,p}$ if $\Pr[X = k] = \binom{n}{k} p^k (1-p)^{n-k}$. We remark that $\mathbb{E}[X] = np$.*

A random variable $Y \in \{0, \ldots\}$ has the Poisson distribution *$\mathcal{P}_\lambda$ if $\Pr[Y = k] = \exp(-\lambda)\lambda^k/k!$. We remark that $\mathbb{E}[Y] = \lambda$. Finally, a random variable $Z \in \{1, \ldots\}$ has the* geometric distribution *$\mathcal{G}_p$ if $\Pr[Z = k] = p(1-p)^{k-1}$.*

Random variables that are “sub-Bernoulli” are a basic analytic element in the theory of blockchains. We define two important species below.

Definition 11 (The martingale condition [2]). *Consider a family of random variables $X_1, \ldots, X_n$ taking values in $\{0, 1\}$. We say that they satisfy the* γ-martingale *condition if, for each $k \geq 1$, $\Pr[X_k = 1 \mid X_1, \ldots, X_{k-1}] \leq 1 - \gamma$.*

The following statement summarizes some classical results in discrete probability theory.

Fact 1. *Let $X_1, \ldots, X_n$ satisfy the $(1-p)$-martingale condition. Let B have the distribution $\mathcal{B}_{n,p}$. Let $\exp(-\lambda) \leq (1-p)^n$ and let P have the Poisson distribution $\mathcal{P}_\lambda$. Then*

1. *$\sum_i X_i \prec B$;*
2. *$B \prec P$;*
3. *for $b \geq \lambda$, $\Pr[P \geq b] \leq \exp(-(b-\lambda)^2/(2b))$, and*
4. *for $b \geq \lambda$, $\Pr[P \geq b] \leq \exp(-\lambda)(e\lambda/b)^b$.*

The first of these tail bounds is looser, but more convenient for one of our settings. Finally, the condition $\exp(-\lambda) \leq (1-p)^n$ is achieved by $\lambda = -n \ln(1-p) = n(p + p^2/2 + \cdots) \leq n(p + p^2)$ (under the mild assumption that $p < 1/2$).

Proof. These are classical results in discrete probability theory. See [5] for a proof of the first, [25] for a proof of the second, and [35,28] for proofs of the tail bounds. □

We recall the fact that if P_1 and P_2 are independent, Poisson distributed random variables with parameters λ_i then $P_1 + P_2$ is also Poisson distributed, with parameter $\lambda_1 + \lambda_2$.

Barrier walks. Central to our analysis are so-called barrier walks. We define them and state some useful properties in the following.

Definition 12 (The barrier walk). *Let $(x_1, \ldots, x_n) \in \mathbb{Z}^n$. We define the values $(w_0, w_1, \ldots, w_n) \in \mathbb{N}^{n+1}$ by the rule $w_0 = 0$, and, for $t > 0$, $w_t = \max(w_{t-1} + x_t, 0)$. Note that these values are given by the height of the natural random walk on the integers given by the values x_i with a barrier at 0. For convenience we write $(w_0, \ldots, w_n) = \mathsf{W}(x_1, \ldots, x_n)$. We extend this notation to act in the obvious way to an infinite sequence $x_1, \ldots$.*

The following lemma expresses the useful fact that shifting mass to the right can only increase the final height of the walk. More formally:

Lemma 3 (The exchange principle for the barrier walk). *Let $(x_1, \ldots, x_n) \in \mathbb{Z}^n$ and let $(w_0, \ldots, w_n) = \mathit{W}(x_1, \ldots, x_n)$. Let $t \in \{1, \ldots, n-1\}$ and define $e_1, \ldots, e_n \in \mathbb{Z}$ so that $e_t = -1$, $e_{t+1} = 1$, and $e_i = 0$ for $i \notin \{t, t+1\}$. Let $x'_i = x_i + e_i$ and let $(w'_1, \ldots, w'_n) = \mathsf{W}(x'_1, \ldots, x'_n)$. Then for every $t > i$, $w'_t \geq w_t$.*

Recall that for a real-valued random variable X, the *moment-generating function* m_X is defined by the rule $z \mapsto \mathbb{E}[e^{zX}]$ (when this expectation exists).

Lemma 4. *Let $R \in \mathbb{N}$; let $\alpha \in (0,1)$ and $C \geq 1$. Let A be a random variable on $\{-R, -R+1, \ldots\}$ satisfying $\mathbb{E}[A] < 0$ and $\Pr[A = k] \leq C\alpha^k$. Then*

$$m_A(\lambda) \leq 1 + \mathbb{E}[A]\lambda/2$$

so long as $\lambda \leq 1/e$ further satisfies

$$\lambda \ln^2(\lambda^{-1}) \leq -\frac{|\,\mathbb{E}[A]|}{8}\left(\frac{\ln(1/\alpha)}{6}\right)^2 , \tag{6}$$

$$\lambda \ln(\lambda^{-1}) \leq \frac{1-\sqrt{\alpha}}{12C} , \quad \textit{and} \tag{7}$$

$$\lambda \leq \sqrt{C}\alpha^{R/4} . \tag{8}$$

Proof. We decompose $m_A(\lambda)$ into two sums:

$$m_A(\lambda) = \underbrace{\sum_{-R \leq k < S} e^{\lambda k} \Pr[A = k]}_{(\dagger)} + \underbrace{\sum_{S \leq k} e^{\lambda k} \Pr[A = k]}_{(\ddagger)} ,$$

where $S = 2 \cdot \ln(\lambda^2(1-\sqrt{\alpha})/C)/\ln(\alpha)$ is a threshold chosen to balance error terms determined below. Anticipating the bounds below, we record the fact that under constraint (7) (and $\ln(\lambda^{-1}) \geq 1$) we have the simpler upper bound

$$S \leq 6\ln(\lambda)/\ln(\alpha) \tag{9}$$

Considering the sum $(\dagger)$, we note that $R \leq S$ follows from constraint (8). Likewise, as $(1-\sqrt{\alpha}) \leq \ln(1/\alpha)$ for all $\alpha \in (0,1)$, we have $S \leq 1/(2\lambda)$ as a result of (9) and constraint (7). We remark that for $|\delta| < 1$, $|\exp(\delta) - (1+\delta)| \leq \delta^2/[2(1-|\delta|)]$ and hence

$$\begin{aligned} &\sum_{-R \leq k < S} e^{\lambda k} \Pr[A = k] \\ &\qquad \leq \sum_{-R \leq k < S} \left(1 + \lambda k + \frac{\lambda^2 k^2}{1 - |\lambda k|}\right) \Pr[A = k] \\ &\qquad \leq 1 + \lambda\,\mathbb{E}[A] + \frac{\lambda^2 S^2}{2(1-\lambda S)} \\ &\qquad \leq 1 + \lambda\,\mathbb{E}[A] + \lambda^2 S^2 . \end{aligned} \tag{10}$$

As for the sum $(\ddagger)$, we note that in light of the constraint (7) we have $\lambda < (1-\sqrt{\alpha})/2 \leq \ln(1/\alpha)/2$ so that $\alpha e^\lambda < \sqrt{\alpha}$. Then we find that

$$\sum_{S \leq k} e^{\lambda k} \Pr[A = k] \leq \sum_{S \leq k} e^{\lambda k} C\alpha^k = \frac{C(\alpha e^\lambda)^S}{1 - \alpha e^\lambda} \leq \frac{C\sqrt{\alpha}^S}{1-\sqrt{\alpha}} \leq \lambda^2 . \tag{11}$$

We conclude that

$$m_A(\lambda) \leq 1 + \lambda\,\mathbb{E}[A] + \lambda^2 S^2 + \lambda^2 \leq 1 + \lambda\,\mathbb{E}[A] + 2\lambda^2 \left(\frac{6\ln(1/\lambda)}{\ln(1/\alpha)}\right)^2 .$$

To ensure that $m_A(\lambda) < 1 + \mathbb{E}[A]\lambda/2$, then, the additional constraint (6) is sufficient. □

Lemma 5. *Let $R \in \mathbb{N}$, $\alpha \in (0,1)$, $C \geq 1$, and $\gamma > 0$. Consider a sequence of integer-valued random variables $Z_1, P_1, Z_2, P_2, \ldots$ satisfying (i.) for each k, $-R \leq Z_k \leq 0$ and $0 \leq P_k$, and (ii.) for each k, for any conditioning on $\{Z_i, P_i \mid i < k\}$,*

$$\mathbb{E}[Z_k + P_k] \leq -\gamma \quad \textit{and} \quad \Pr[Z_k + P_k = t] \leq C \cdot \alpha^t \,. \tag{12}$$

We prove two tail bounds involving such variables:

(1) (The barrier tail.) Let $(0, \tilde{W}_1, W_1, \tilde{W}_2, W_2, \ldots) = \mathsf{W}(Z_1, P_1, Z_2, P_2, \ldots)$. Then there are constants $\alpha_b, C_b > 0$ so that for all $n \geq 0$,

$$\forall T, \quad \Pr[W_n \geq T] \leq C_b e^{-\alpha_b T} \,. \tag{13}$$

(2) (The free tail.) Let $S_n = \sum_{i=1}^n Z_i + P_i$. Then there is a constant $\alpha_f > 0$ so that

$$\forall T \geq -\gamma n/2, \quad \Pr[S_n \geq T] \leq e^{-\alpha_f (T + \gamma n/2)} \,. \tag{14}$$

Proof. We begin with the barrier tail (13) above. We will establish, by induction on n, that for some constants $C_b, \alpha_b > 0$, the moment generating functions $m_n(z) = \mathbb{E}[\exp(z W_n)]$ satisfy $m_n(\alpha_b) \leq C_b$. In that case,

$$\Pr[W_n \geq T] = \Pr[e^{\alpha_b W_n} \geq e^{\alpha_b T}] \leq \frac{\mathbb{E}[e^{\alpha_b W_n}]}{e^{\alpha_b T}} \leq C_b e^{-\alpha_b T} \,.$$

We return to the task of bounding above the moment generating functions $m_n(z)$ as described above. Let $\lambda^* > 0$ be the minimum value of the three constraints (6)–(8) demanded by Lemma 4 for R, α, C, and (expectation) $-\gamma$. Then any random variable D satisfying the conditions (12) above has a moment-generating function for which $m_D(\lambda^*) \leq 1 - \gamma\lambda^*/2$. To complete the proof, we establish by induction that $m_n(\lambda^*) \leq (2/(\lambda^*\gamma) - 1)e^{\lambda^* R}$; as this value is independent of n, this completes the proof. The base case, $n = 0$, follows as $2/(\lambda^*\gamma) - 1 > 1$ (because $\mathbb{E}[D] \geq -R$ and, in the proof of Lemma 4 we take $\lambda \leq 1/(2S) \leq 1/(2R)$) and the moment generating function of W_0 is everywhere 1. For the inductive case, we expand

$$\begin{aligned} m_n(\lambda^*) &= \Pr[W_{n-1} \geq R] \cdot \mathbb{E}[e^{\lambda^* W_n} \mid W_{n-1} \geq R] + \Pr[W_{n-1} < R] \cdot \mathbb{E}[e^{\lambda^* W_n} \mid W_{n-1} < R] \\ &\leq \mathbb{E}[e^{\lambda^*(W_{n-1} + Z_n + P_n)}] + \mathbb{E}[e^{\lambda^* W_n} \mid W_{n-1} < R] \\ &\leq \mathbb{E}[e^{\lambda^* W_{n-1}} \, \mathbb{E}[e^{\lambda^*(Z_n + P_n)} \mid W_{n-1}]] + \mathbb{E}[e^{\lambda^*(R + Z_n + P_n)} \mid W_{n-1} < R] \\ &\leq (1 - \lambda^*\gamma/2) \, \mathbb{E}[e^{\lambda^* W_{n-1}}] + e^{\lambda^* R} \, \mathbb{E}[e^{\lambda^*(Z_n + P_n)} \mid W_{n-1} < R] \\ &\leq (1 - \lambda^*\gamma/2)(e^{\lambda^* R} + m_{n-1}(\lambda^*)) \,. \end{aligned} \tag{15}$$

In the expansion above, we critically use the fact that when $W_{n-1} \geq R$ the value W_n is simply the sum $W_{n-1} + Z_n + P_n$. Combining the hypothesis $m_{n-1}(\lambda^*) \leq (2/(\lambda^*\gamma) - 1)e^{\lambda^* R}$ with (15) results in the identical upper bound for $m_n(\lambda^*)$, as desired.

Finally, we address the free tail (14). Let $D_k = Z_k + P_k$; then, as above, $m_{D_k}(\lambda^*) \leq 1 - \lambda^*\gamma/2 \leq \exp(-\lambda^*\gamma/2)$. Expanding the moment generating function of $\sum D_i$ with the observation that

$$\mathbb{E}[e^{\lambda^* \sum_i^k D_i}] = \mathbb{E}[e^{\lambda^* \sum_i^{k-1} D_i} \, \mathbb{E}[e^{\lambda^* Z_k} \mid D_1, \ldots, D_{k-1}]] \leq \mathbb{E}[e^{\lambda^* \sum_i^{k-1} D_i}] m_{D_k}(\lambda^*)$$

yields the bound $m_{\sum D_i}(\lambda) \leq \exp(-\lambda^*\gamma n/2)$. As discussed in the opening section of this proof, this yields the bound $\Pr[\sum_i D_i \geq T] \leq e^{\lambda^*(T + \gamma n/2)}$, as desired. □

Distribution of the characteristic string. In the PoW setting, during an adversarial spike we will encounter characteristic strings $W = W_1, \ldots W_\ell \in \mathcal{W}_{\mathsf{pow}}^\ell$ where each symbol is distributed according to some distribution $\mathcal{D}_p(N_h^{(i)}, N_a^{(i)})$ from the family given in Definition 8 with some $p < 1/2$. The parameters $N_h^{(i)}$ and $N_a^{(i)}$ are random variables chosen by the adversary and are restricted to satisfy $N_a^{(i)} \leq (1 - \epsilon) \cdot \theta_{\mathsf{pow}} \cdot N_h^{(i)} + G_i$ for constant $\varepsilon > 0$ and some threshold θ_{pow} determined below, where random variables $G_i \in \mathbb{N}$ must obey $\sum_{i=1}^\ell G_i \leq \mathcal{B}$ with probability 1.

Our intermediate goal is to understand the distribution of $\beta_\ell^\Delta(W)$. In the following analysis, we will first study the value $\beta_\ell^\Delta(\widehat{W})$ for $\widehat{W} = \widehat{W}_1 \dots \widehat{W}_\ell$ with each $\widehat{W}_i$ distributed according to $\mathcal{D}_p(N_h^{(i)}, \widehat{N}_a^{(i)})$, where $\widehat{N}_a^{(i)} \triangleq \max\{0, N_a^{(i)} - G_i\}$. Hence, these parameters satisfy $\widehat{N}_a^{(i)} \leq (1-\epsilon) \cdot \theta_{\mathsf{pow}} \cdot N_h^{(i)}$ and, intuitively, represent the behavior of $\beta_\ell^\Delta()$ under sufficient honest majority, without the effect of the adversarial spike. Afterwards, we extend the analysis to also take into account the spike represented by the additional G_i adversarial hashing attempts in each round i.

The exact PoW threshold required follows roughly speaking from the requirement that, without the adversarial spikes, the expected number of isolated uniquely honest rounds should dominate the expected number of adversarial mining successes (recall that a round i is uniquely honest if $\widehat{W}_i = (1, \cdot)$). Round i is isolated uniquely honest with probability

$$N_h^{(i)} p (1-p)^{N_h^{(i)}-1} \cdot \prod_{|i-j| \in [\Delta]} (1-p)^{N_h^{(j)}} = N_h^{(i)} \cdot \frac{p}{1-p} \cdot \prod_{|i-j| \leq \Delta} (1-p)^{N_h^{(j)}} .$$

A simple threshold can be obtained by defining

$$\theta_{\mathsf{pow}} = \theta_{n,p,\Delta} \triangleq (1-p)^{(2\Delta+1)n} \tag{16}$$

and observing that this implies for our settlement game that $(N_h^{(i)}, N_a^{(i)})$ are therefore (adaptively) chosen by the adversary so that for $\xi > 0$ and all $i \in [\ell]$ it follows that

$$N_h^{(i)} \cdot \frac{p}{1-p} \cdot \prod_{|i-j| \leq \Delta} (1-p)^{N_h^{(j)}} > \theta_{n,p,\Delta} \cdot N_h^{(i)} \cdot p \geq (1+\xi) \cdot (N_a^{(i)} - G_i) \cdot p \tag{17}$$

with probability 1. Note that the right-hand side clearly upper-bounds the expected number of adversarial successes in round i without the effect of the spike, while the threshold on the left takes the role of a discount factor of the honest mining power (seen as the honest hashing units times probability of success). Note that similar conditions were assumed also in previous analyses [24,37,33,3], where often the linear approximation by Bernoulli's inequality is taken, i.e., $\theta_{n,p,\Delta} \geq 1 - 2(\Delta+1)np$ would be applied in equation (17).

Additionally, we will also assume that the number of active parties $N_h^{(i)} + \widehat{N}_a^{(i)}$ in each round i is bounded by constants $n_0, n \in \mathbb{N}$: we have

$$n_0 \leq N_h^{(i)} + \widehat{N}_a^{(i)} \leq n \tag{18}$$

with probability 1.

We now note some properties of any $\widehat{X} \in \rho_\Delta(\widehat{W})$. We write $\widehat{X} = \widehat{X}_1 \dots \widehat{X}_\ell$, where each $\widehat{X}_i \in \{\mathsf{h}, \mathsf{a}\}^*$ comes from $\mathcal{X}_i$ as defined in (5) with respect to $\widehat{W}$. To be able to invoke Lemma 5, our analysis will look at longer sequences of symbols within $\widehat{W}$, and consequently $\widehat{X}$. To this end, fix some

$$m \triangleq c_m \cdot \Delta \quad \text{for} \quad c_m \in \mathbb{N},\ c_m \geq 2 \ ; \tag{19}$$

and assume for simplicity that $m|\ell$. Now for $i \in \{1, \dots, \ell/m\}$ define

$$Z_i \triangleq -\#_{\mathsf{h}}(\widehat{X}_{(i-1)m+\Delta+1} \dots \widehat{X}_{im}) \quad \text{and} \quad P_i \triangleq \#_{\mathsf{a}}(\widehat{X}_{(i-1)m+1} \dots \widehat{X}_{im}) \ . \tag{20}$$

We now show that under a suitable parametrization, these random variables satisfy the preconditions of Lemma 5.

Lemma 6. *Let $\widehat{W} = \widehat{W}_1, \dots, \widehat{W}_\ell$ be as above, satisfying* (17) *and* (18)*. Let $\widehat{X} = \widehat{X}_1 \dots \widehat{X}_\ell \in \rho_\Delta(\widehat{W})$ be as above. Then the random variables (Z_i, P_i) of* (20) *with $c_m = \lceil 1 + 2/\xi \rceil$ satisfy the preconditions of Lemma 5 with*

$$\gamma = (\Delta/(2+2\xi)) n_0 p (1-p)^{(2\Delta+1)n} \ , \quad C = \exp((p+p^2) \cdot n) \quad \textit{and} \quad \alpha = e^{-1/2} \ . \tag{21}$$

Proof. Clearly $P_k > 0$ and $-R \leq Z_k \leq 0$ for R being an upper bound on the number of hash queries issued in rounds $(k-1)m+1, \ldots, km$ (recall that the number of parties is upper-bounded in our model). Fix any $k \in [\ell]$ and any conditioning on $\{Z_i, P_i \mid i < k\}$ for the rest of the proof, it remains to show that: (a) $\mathbb{E}[Z_k + P_k] \leq -\gamma$; (b) $\Pr[Z_k + P_k = t] \leq C \cdot \alpha^t$.

Observe that by definition of ρ_Δ, $\#_{\mathsf{h}}(\widehat{X}_k) = 1$ if and only if k is an isolated, uniquely honest index in $\widehat{W}$; otherwise $\#_{\mathsf{h}}(\widehat{X}_k) = 0$. Moreover, if we write $\widehat{W}_i = (\widehat{H}_i, \widehat{A}_i) \in \mathcal{W}_{\mathsf{pow}}$, then we have $\#_{\mathsf{a}}(\widehat{X}_k) = \widehat{A}_k$ as the set $\mathcal{X}_k$ from (5) only contains strings with exactly $\widehat{A}_i$ occurrences of the symbol a.

Towards proving (a), note that thanks to omitting the variables $\widehat{X}_{(i-1)m+1}, \ldots, \widehat{X}_{(i-1)m+\Delta}$ in the definition of Z_i in (20), for any $\widehat{X}_j$ affecting Z_k (i.e., such that $j \in \{(k-1)m+\Delta+1, \ldots, km\}$) we have that, even conditioned on $\{Z_i, P_i \mid i < k\}$,

$$e_h \triangleq \mathbb{E}\left[\#_{\mathsf{h}}(\widehat{X}_j)\right] = N_h^{(j)} \cdot \frac{p}{1-p} \cdot \prod_{|j'-j| \leq \Delta} (1-p)^{N_h^{(j')}} \tag{22}$$

and

$$e_a \triangleq \mathbb{E}\left[\#_{\mathsf{a}}(\widehat{X}_j)\right] = p \cdot \widehat{N}_a^{(j)} \ .$$

Assumption (17) implies $e_h > (1+\xi)e_a$ and hence for our choice of $c_m = \lceil 1 + 2/\xi \rceil$ we get

$$\begin{aligned}\mathbb{E}\left[Z_k + P_k\right] &= me_a - (m-\Delta)e_h = \Delta e_a - \Delta(c_m - 1)(e_a - e_h) < e_h \Delta \left(\frac{1}{1+\xi} - (c_m - 1) \cdot \frac{\xi}{1+\xi}\right) \\ &\leq -(\Delta/(1+\xi)) \cdot e_h \ .\end{aligned}$$

Assumption (18) together with (17) implies $n_0/2 \leq N_h^{(j)} \leq n$, hence (22) gives us

$$e_h \geq \tilde{e}_h \triangleq (1/2) n_0 p (1-p)^{(2\Delta+1)n} \ , \tag{23}$$

establishing statement (a) with γ as in (21).

Statement (b) follows from $\Pr[Z_k + P_k = t] \leq \Pr[P_k \geq t]$ and from Fact 1, as the distribution of each $\widehat{A}_k$ is stochastically dominated by a Poisson distribution $\mathcal{P}_{\lambda_k}$ with $\lambda_k = (p + p^2) \cdot \widehat{N}_a^{(k)}$. □

Putting things together. We now study the situation where, informally speaking, in an execution of a Nakamoto-style PoW blockchain, an adversarial spike bounded by a budget $\mathcal{B}$ occurs before (and up to) some round ℓ. We upper-bound the probability that after round ℓ, after a further waiting period D has passed, the quantity β_ℓ^Δ has not yet reached back 0.

Lemma 7. *Fix some $p \in (0, 1/2)$, integers $\ell, D > 0$ and denote $L = \ell + D$. Let $W = W_1 \ldots, W_L \in \mathcal{W}_{\mathsf{pow}}^L$ be a family of random variables, each W_i sampled according to $\mathcal{D}_p(N_h^{(i)}, N_a^{(i)})$ with parameters satisfying (17)* and (18). *Moreover, assume that $\sum_{i=1}^{\ell} G_i = \mathcal{B}$ and $G_i = 0$ for each $i > \ell$ with probability* 1. *Then there exists some $D_0 = O(p\mathcal{B})$ such that if $D > D_0$ then we have*

$$p_{\mathsf{bad}} \triangleq \Pr\begin{bmatrix} \forall i \in \{\ell+1, \ldots, L\}\colon \\ \beta_\ell^\Delta(W_{1:i}) > 0 \end{bmatrix} \leq \exp(-\Omega(D)) \ .$$

Note that in the above statement $O(p\mathcal{B}) = O(\mathcal{B})$ as p is a constant; we keep it as a part of the presentation to emphasize that the relevant quantity here is the number of adversarial successes which is $p\mathcal{B}$ on expectation.

Proof sketch. Let (X, Y) be the element of $\rho_\ell^\Delta(W)$ maximizing $\beta_{|X|}(XY)$, let $\ell' \triangleq |X|$ and $L' \triangleq |XY|$ (note that these are all random variables determined by W). We will investigate $\beta_{\ell'}(XY_{1:i})$, as Lemma 2 gives us

$$p_{\mathsf{bad}} \leq \Pr_W \begin{bmatrix} \forall i \in \{1, \ldots, L' - \ell'\} \\ \beta_{\ell'}(XY_{1:i}) + \mathsf{mres} > 0 \end{bmatrix}$$

where $\mathsf{mres} \triangleq \max_{i\in[D]} \mathrm{res}_\Delta(W_{1:\ell+i})$. As mres is constant in both $\mathcal{B}$ and D, we ignore its effect in the discussion below for the sake of simpler notation, accounting for it would be straightforward.

We consider a sequence of random variables $\widehat{W} = \widehat{W}_1 \dots, \widehat{W}_L$ distributed according to $\mathcal{D}_p(N_h^{(i)}, \widehat{N}_a^{(i)})$ as defined above and naturally coupled with W. Let $(\widehat{X}, \widehat{Y}) \in \rho_\ell^\Delta(\widehat{W})$ corresponding to (X, Y), let $\widehat{\ell}' \triangleq |\widehat{X}|$, and decompose $\widehat{X}$ as $\widehat{X} = \widehat{X}_1 \dots \widehat{X}_\ell$, where each $\widehat{X}_i \in \{\mathsf{h}, \mathsf{a}\}^*$ comes from $\mathcal{X}_i$ as defined in (5).

To first understand $\beta_{\widehat{\ell}'}(\widehat{X})$, we invoke Lemma 5 with Z_i and P_i defined as in (20). Recall that Z_i and P_i satisfy the preconditions of Lemma 5 by Lemma 6. Since $\beta_{\widehat{\ell}'}(\widehat{X})$ performs a barrier walk based on $\widehat{X}$ as established in Lemma 1, we can conclude based on statement (1) of Lemma 5 that for any t,

$$\Pr[\beta_{\widehat{\ell}'}(\widehat{X}) \geq t] = \exp(-\Omega(t)) .$$

Now we move to $\beta_{\ell'}(X)$, i.e., we account for the effect of the adversarial spike. For the parameters $\widehat{N}_a^{(i)}$ and $N_a^{(i)}$ defining the distributions of $\widehat{W}_i$ and W_i respectively, we have $N_a^{(i)} - \widehat{N}_a^{(i)} \leq G_i$ and hence the sampling of W_i differs by at most G_i additional adversarial "PoW attempts." The number of successes S_i coming from these attempts will be binomially distributed for each i, hence by Fact 1 the distribution of $S \triangleq \sum_{i\in[\ell]} S_i$ will be stochastically dominated by $\mathcal{P}_\lambda$ for $\lambda = \mathcal{B} \cdot (p + p^2)$ and will satisfy $P[S \geq \lambda + t] \leq \exp(\lambda/2 - \Omega(t))$. By definition of ρ_Δ, this will result in S additional "a"-symbols in X compared to $\widehat{X}$, and by Lemma 3 these can be moved to the end of the string to quantify the effect on $\beta_{\ell'}$, i.e., on this barrier walk (recall Lemma 1 with $\ell' = |X|$). Put together, we obtain

$$\Pr[\beta_{\ell'}(X) \geq 2p\mathcal{B} + t] \leq \exp(p\mathcal{B} - \Omega(t)) . \tag{24}$$

Finally we study $\beta_{\ell'}(XY_{1:i})$ for $i \geq 1$. This value follows a simple negatively-biased random walk by Lemma 1 (except for the short Δ-suffix of Y that affects the outcome of the walk by at most Δ), hence the probability that it remains positive for all $i \in [|Y|]$ is upper-bounded by the probability that $\beta_{\ell'}(XY) > 0$. To bound this probability, we consider variables $\{Z_i, P_i\}_{i=1}^{D/m}$ defined in the same way as in (20), based on the string Y instead of $\widehat{X}$; there will be D/m such variables. By Lemma 6 and the fact that $\sum_{i>\ell} G_i = 0$, these variables again satisfy the preconditions of Lemma 5 and we can invoke its statement (2) to obtain

$$\Pr\left[\beta_{\ell'}(X) - \beta_{\ell'}(XY) \leq \frac{\gamma D}{3m}\right] \leq \exp(-\Omega(D)) \tag{25}$$

for γ and m as in (21) and (19), respectively. Combining equations (24) and (25) yields the lemma. □

The following theorem summarizes our main result for PoW blockchain consistency.

Theorem 1. *The Nakamoto-style PoW blockchain executed with at most n participants over a network with maximum delay Δ, PoW success probability p, and cut-off parameter κ satisfies the consistency self-healing property against any $(\theta_{n,p,\Delta}, \epsilon, n_0, \mathcal{B})$-adversary, where the threshold is defined in equation* (16), *and where $\epsilon, n_0 > 0$ are arbitrary constants, with a vulnerability period given by the pair (τ_l, τ_h) where $\tau_l = O(\mathcal{B})$ and $\tau_h = O(\mathcal{B}) + O(\kappa)$.*

Proof sketch. Consider the event $\mathsf{ConsFail}(r_1, r_2)$ and for the sake of analysis, let ℓ denote the round in which the κ-deep block in the chain held by P_1 in round r_1 was mined, we will naturally be interested in the quantity β_ℓ^Δ whose connection to consistency (violations) is explained in Section 5.3. Recall that ρ_a and ρ_b denote the first and respectively last round in which the adversary exceeds his budget. According to Def. 3, we need to consider two cases: $r_1 \geq \rho_b + \tau_h$ and $r_1 \leq \rho_a - \tau_l$.

To be able to argue about the case $r_1 \geq \rho_b + \tau_h$, we choose τ_h so that (i) $\ell \geq \rho_b$, this requirement can be met with a τ_h satisfying $\tau_h = O(\kappa)$ thanks to the well-known linear chain growth property exhibited by this class of protocols (see Appendix A for a survey); and (ii) $\tau_h \geq D_0 = O(\mathcal{B})$ from Lemma 7. The statement then follows from Lemma 7 which establishes that after τ_h rounds since the end of an adversarial spike have passed, the quantity β_ℓ^Δ hits 0 with overwhelming probability, returning the probabilities of a consistency violation to being identical to a spike-free execution. Note that while Lemma 7 directly analyzes the case $\ell = \rho_b$ where immediately after the spike ends, the biased random walk performed by β_ℓ^Δ changes from barrier to free, the results also apply to the case $\ell > \rho_b$ as the barrier and free random walks behave identically on positive values until the walk hits the barrier 0 for the first time.

For $r_1 \le \rho_a - \tau_l$, the situation is somewhat simpler. The quantity β_ℓ^Δ performs a negatively-biased barrier walk up to round ℓ, and a negatively-biased free random walk after ℓ. The effect of the spike (which occurs at least τ_l rounds after ℓ) can be—according to Lemma 3—accounted for by requiring that β_ℓ^Δ drops below $\mathcal{B}$ instead of 0. For a proper choice of $\tau_l = O(\mathcal{B})$, this occurs up to round $\ell + \tau_l$ with overwhelming probability by an argument similar to Lemma 7. Therefore, when the adversarial spike occurs, the value β_ℓ^Δ will be sufficiently negative to ensure that the assumed spike cannot make it reach non-negative values again. □

5.4 Explicit Bounds on Recovery from Adversarial Majority

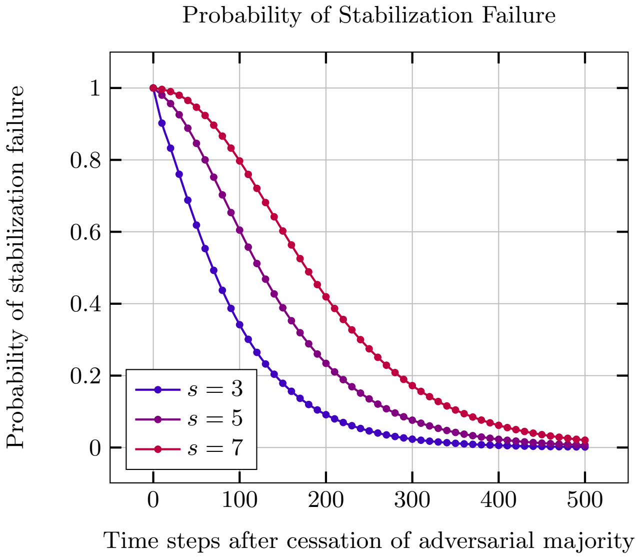

Fig. 2. Graphs of the probability that a PoW blockchain fails to stabilize after a period of adversarial majority. These plots correspond to a 10% adversary, $\Delta = 3$, and a success probability p chosen so that the expected number of honest successes over $7 = 2\Delta + 1$ rounds is 1. The three graphs show adversarial quotas corresponding to 3, 5, and 7

These exact numerical calculations reflect a setting with a large number of participants and (comparably) small PoW success probability. Specifically, for large N we assign $n_h = .9 * N$, $n_a = .1 * N$, $\Delta = 3$ and $p = 1/(7n_h)$. Observe that the expected number of honest successes in a single round is 1/7 and the expected number of successes over $7 = 2\Delta + 1$ rounds is 1. The exact probability of a uniquely honest, doubly-isolated round depends on n_h; however, when N is large, the distribution of honest successes in a single slot is tightly approximated by the Poisson distribution with parameter 1/7 for which the probability of a uniquely honest doubly isolated round is $(1/7)e^{-1/7} \cdot \left(e^{-1/7}\right)^6 = (1/7)e^{-1} = 0.0525547...$. We use the Poisson distribution of successes—for both honest and adversarial participants—in the numerical simulations. In contrast, the expected number of adversarial successes in a single round is $1/(7 * 9) = 0.015873\ldots$.

Interpreting the adversarial spike budget. The adversarial spike budget is denoted by s and can be understood as the expected number of "extra" adversarial PoW hash successes available to the adversary during the spike. More concretely, $s = 1$ corresponds to any spike where the extra hashing power given to the adversary is enough to produce (in expectation) one more hashing success. For the setting above (where

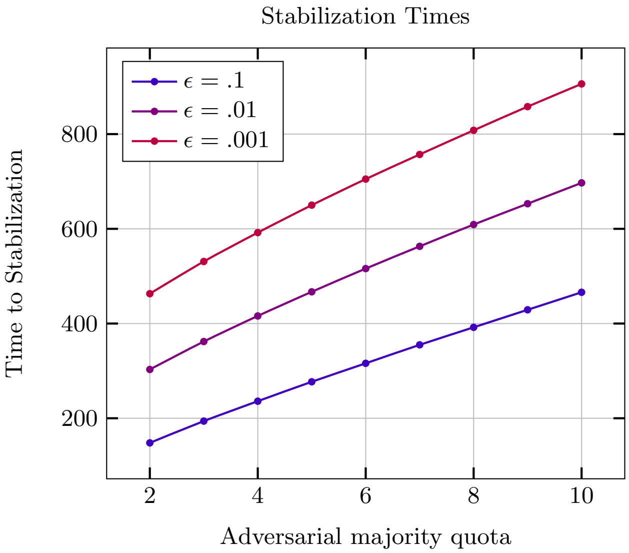

Fig. 3. Graphs of the time necessary for a PoW blockchain to stabilize (with prescribed error) with various spike quotas. These plots correspond to a 10% adversary, $\Delta = 3$, and a success probability p chosen so that the expected number of honest successes over $7 = 2\Delta + 1$ rounds is 1. The three graphs show target errors of 1/10, 1/100, and 1/1000 over a range of adversarial spike quotas.

the adversary has in expectation one success in 63 rounds), a spike budget of $s = 3$ can be understood—for example—as doubling the power within a time interval of 189 rounds.

This suggests an alternate mechanism to describe spike power consisting of identifying the temporarily increased *fraction of total hashing power* held by the adversary and the duration of the spike. For example, continuing to adopt the nominal 10% adversary above (with $n_a = .9N$ and $n_h = .1N$), consider a period of 19 rounds during which the adversary is provided with $n_s = .945N$ extra hashing power. During this time, the adversary comprises $53.8\% = (n_a + n_s)/(n_s + n_a + n_h)$ of all hashing power and generates, in expectation, $s = 3.0...$ additional PoW successes.

Similar calculations (all adopting the same nominal conditions) yield the table below, which give various (hashingpower) × (duration) equivalences for various quotas. The column labeled "spike power" indicates the fraction of all hashing power held by the adversary during the spike. The values are chosen to reflect two settings of interest: an adversary with a slight majority ($\sim 53\%$) and one with a more significant majority ($\sim 69\%$).

Quota	Equivalent spike	
(s)	Spike power	Spike duration (rounds)
3	53.7%	20
5	53.9%	33
7	54 %	46
3	68.5%	10
5	68.5%	17
7	69.2%	23

The recovery bounds are further graphically illustrated in Figures 2, 3, and 4.

The source code of the simulation of the random walks and the computation of the probability distributions are available at the public github repository `https://github.com/russella/blockchain-spike-numerics`.

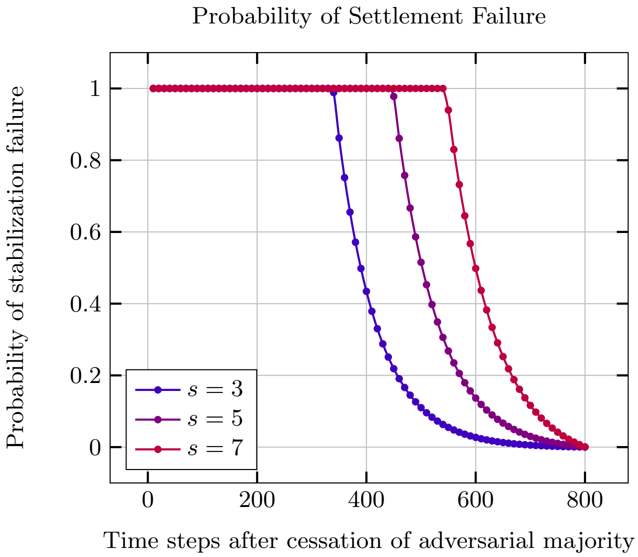

Fig. 4. Graphs of the probability that a PoW blockchain fails to settle after a period of adversarial majority. These plots correspond to a 10% adversary, $\Delta = 3$, and a success probability p chosen so that the expected number of honest successes over $7 = 2\Delta + 1$ rounds is 1. The three graphs show adversarial quotas of 3, 5, and 7.

6 Nakamoto-Style Proof of Stake

For concreteness, we choose the Ouroboros Genesis protocol [2] as an instantiation of a Nakamoto-style PoS protocol for which we express our results. As fewer readers might be familiar with this protocol compared to Bitcoin, in the following we include its high-level description.

6.1 High-Level Description of Ouroboros Genesis

The protocol operates and was analyzed in the semi-synchronous model with fully adaptive corruptions.

Basic operation. In each round, each of the parties determines whether she qualifies as a so-called *leader* for this round. This is done by locally evaluating a verifiable random function (VRF) [12] using the secret key associated with their stake, and providing as inputs to the VRF both the round index and so-called *epoch randomness* (we will discuss shortly the source of this randomness). If the VRF output is below a certain threshold—proportional to the party's stake as detailed below—then the party is an eligible leader and she creates a block for that round extending the chain she currently holds, signs and broadcasts it. The block contains transactions that move stake among stakeholders; the above VRF output and its proof to certify leader eligibility; and also another, independent VRF value with a proof to be used for determining the next epoch randomness. Note that the event of a particular party becoming a leader is independent for two different rounds and two different parties, leading to some rounds with no, or several, leaders.

Parties participating in the protocol continuously collect such valid blocks and update their current state to reflect the longest chain they have observed so far that does not fork from their previous state by too many blocks into the past. This safety depth is a parameter of the protocol.

Iterative epoch structure. Multiple rounds are combined into *epochs*, each of which contains $R \in \mathbb{N}$ rounds. The epochs are indexed by $j \in \mathbb{N}$. During epoch j, leader election is based on the stake distribution $\mathbb{S}_j$ recorded in the blockchain up to g rounds before the beginning of this epoch (in [2,11] the value $g = R$ is chosen). The *epoch randomness* for epoch j is derived as a hash of the additional VRF-values that were included into blocks up to h rounds before the beginning of this epoch (in [2,11] the value $h = R/3$ is chosen).

Newly joining parties. What distinguishes Genesis from its predecessor Ouroboros Praos [11] is a secure procedure for parties to trustlessly join the execution later, called the *Genesis rule*: Joining parties determine their state by listening to the broadcast chains for sufficiently long and applying a specific rule to choose the one they adopt as their state. Namely, for any pair of two competing chains, if they branch in very recent past the longer one is preferred; but if they branch in more distant past, a joining party gives preference to a chain that is more dense (i.e., contains more blocks) in a fixed period of s rounds after the branching point of these two chains (where s is a protocol parameter).

Leader election. The threshold T_i^j for a party P_i, which her VRF output must fall below for P_i to become a leader in a fixed round of epoch j is

$$T_i^j \triangleq 2^{\ell_{\mathsf{VRF}}} \phi_f(\alpha_i^j) \tag{26}$$

where $\alpha_i^j \in [0,1]$ is the relative stake of stakeholder P_i in the stake distribution $\mathbb{S}_j$, ℓ_{VRF} denotes the output length of the VRF, $f \in (0, 1/2]$ is the so-called *active rounds coefficient*—a protocol parameter, and ϕ_f is the mapping

$$\phi_f(\alpha) \triangleq 1 - (1-f)^{\alpha} \ . \tag{27}$$

The function $\phi_f(.)$ that determines the leadership lottery has some useful properties; it is sub-additive and has the independent aggregation property [2,11] which are crucial to analyze the leadership process. We mention two of the core consequences here (and more will be discussed throughout this section). The probability p that there is no slot leader in a round satisfies $1 - f \leq p \leq 1 - \beta f$, where β denotes the fraction of stake that is active in this round. Second, the probability that exactly one participant is elected as slot leader and that this participant is honest can be lower bounded by $\phi_f(\alpha_H)(1-f) \geq \alpha_H f(1-f)$, where α_H denotes the fraction of stake active in this round. This lower bound is independent of how the coins are allocated among active honest parties.

Finally, the protocol uses *key-evolving signatures* (KES) for block signing, and in each round the honest parties update their private key, contributing to their resilience to adaptive corruptions.

In line with [2,11,5], our following analysis focuses on a single-epoch execution of Ouroboros and the inductive lifting to multiple epochs is discussed in Section 6.5.

6.2 Instantiating the Abstract Ledger Model for PoS

The general structure of the protocol execution in a Nakamoto-style PoS protocol is very similar to that of PoW in the previous section. First of all, the elements in this case are denoted $(\mathbb{P}_{\mathsf{pos}}, \Sigma_{\mathsf{pos}}, \mathcal{W}_{\mathsf{pos}}, \mathcal{D}_{\mathsf{pos}}, \mathcal{G}_{\mathsf{pos}}, \dashrightarrow_{\mathsf{pos}}, \mathbb{L}_{\mathsf{pos}})$. We define (i) the set $\mathbb{P}_{\mathsf{pos}}$ to be an arbitrary name space to distinguish honest protocol participants, (ii) $\Sigma_{\mathsf{pos}} \subseteq [0,1]^{\mathbb{P}_{\mathsf{pos}}} \times [0,1]^{\mathbb{P}_{\mathsf{pos}}}$ to be the set that describes all the honest and adversarial parties and their associated stake that may be active in a round and such that the total sum of stake ratios is less than or equal to one (note that we will use the non-flat model in this section cf. Remark 1), (iii) $\mathcal{W}_{\mathsf{pos}} = \mathbb{N} \times \mathbb{N}$ to denote the number of slot leaders for honest and adversarial entities in a given round (which we are going to simplify in the next paragraph). Next, (iv) $\mathcal{D}_{\mathsf{pos}}$ is the distribution as induced by the slot-leader election procedure defined above (cf. equation 26), and (v) $\mathbb{L}_{\mathsf{pos}}(G, P)$ outputs the sub-path of $G|_P$ that corresponds to all blocks that were created at least k rounds ago, capturing the fact that parties in Nakamoto-style PoS blockchain protocols chop off a certain number of blocks in the suffix of the blockchain they possess to arrive at a common prefix; we express this chop-off parameter in terms of number of rounds (recall that blocks reliably report round numbers in PoS).

Simplifications and the PoS Fork. Similar to the PoW case, it will be convenient in the analysis to focus on a simplified version of the execution graph. First, the "good event" we are after is that we have exactly one honest slot leader in a round. Second, the actual number of adversarial slot leaders is not relevant for Nakamoto-style PoS: once a slot leader, the adversary can create any number of blocks in a given round that are valid in the view of distinct honest parties. Hence, in the PoS setting we can use a very simplified alphabet for our characteristic strings, namely $\mathcal{W}_{\mathsf{pos}} \triangleq \{0, \mathsf{h}, \mathsf{a}\}$ (whose associated distribution can be characterized by a general martingale condition as derived later; cf. equation (31)). Following [11], the symbols have the following intuitive meaning: 0 represents a round with no eligible leaders; h indicates that the respective round has a unique, honest leader; and finally a denotes all other cases (i.e., either at least one adversarial leader or multiple honest ones). This leads to an even simpler representation of the execution graph and its transition relation (i.e., elements $\mathcal{G}_{\mathsf{pos}}$ and $\dot{\to}_{\mathsf{pos}}$) in the PoS case compared to the PoW case. The notion of a PoS fork defined in [23,11] captures both of these aspects in one definition and we restate that definition below adapted to our notation. Compared to the PoW fork, we can work with an empty type-label, since $w_i = \mathsf{h}$ indicates that the only block that exists in round i must be honest. Based on the above, the following definition of a fork will capture all the relevant execution graph structure and evolution from one round to the next that we will need for the analysis of the PoS case:

Definition 13 (PoS Δ-fork [23,11]). *Let Δ and L be positive integers, let $w \in (\mathcal{W}_{\mathsf{pos}})^L$ be a word over $\mathcal{W}_{\mathsf{pos}}$. Let $A(w) = \{i \in [L] \mid w_i \neq 0\}$ and $H(w) = \{i \in [L] \mid w_i = \mathsf{h}\} \subseteq A(w)$. A* PoS Δ-fork *for the string w is a directed, rooted tree $F = (V, E)$ with a labeling*

$$\mathsf{l}_\# : V \to \{0\} \cup A(w)$$

satisfying the axioms below. Edges are directed "away from" the root so that there is a unique directed path from the root to any vertex. The value $\mathsf{l}_\#(v)$ is referred to as the label *of v; if $\mathsf{l}_\#(v) \in H(w)$ then v is called* honest*, otherwise it is called* adversarial.

(i) the root $r \in V$ is given the label $\mathsf{l}_\#(r) = 0$;
(ii) the sequence of labels along any (directed) path is increasing;
(iii) each index $i \in H(w)$ is the label of exactly one vertex of F;
(iv) for any pair of honest vertices v, w for which $\mathsf{l}_\#(v) + \Delta \leq \mathsf{l}_\#(w)$, $\mathsf{len}(v) < \mathsf{len}(w)$, where $\mathsf{len}()$ denotes the depth of the vertex.

A reader familiar with the treatment of Δ-forks in [11,2] will observe that our PoS Δ-forks are identical to these objects, with our symbols 0, h, and a in $\mathcal{W}_{\mathsf{pos}}$ exactly matching the meaning of symbols $\perp$, 0, and 1 in the characteristic strings considered there, respectively. We opt for different symbols to highlight the relationship to the proof-of-work case treated in this paper. Also, the Definitions 5 to 7 on notation, tines, and dominance, apply equally to PoS forks.

6.3 Asymptotic Single-Epoch Analysis

Reach and relative margin. We recall the notions of reach and margin defined in [23] and generalized in [5]; we adapt them to our notation and further extend them to the Δ-synchronous setting.

Definition 14 (Reach). *Let $F \vdash_\Delta w$ be a closed fork and let $\widehat{T}$ denote any tine of maximal length in F. We define the* gap *of a tine T, denoted* $\mathrm{gap}(T)$*, to be the difference in length between $\widehat{T}$ and T; thus* $\mathrm{gap}(T) \triangleq \mathsf{len}(\widehat{T}) - \mathsf{len}(T)$*. We define the* reserve *of a tine T in F to be the number of adversarial indices appearing in w after the last index in T; specifically, if T is given by the path $(r, v_1, \ldots, v_k)$, where r is the root of F, we define* $\mathrm{reserve}_F(T) \triangleq |\{i \mid w_i = \mathsf{a} \text{ and } i > \mathsf{l}_\#(v_k)\}|$*. We then define* $\mathrm{reach}_F(T) \triangleq \mathrm{reserve}_F(T) - \mathrm{gap}_F(T)$*,*

$$\rho(F) \triangleq \max_{T \text{ in } F} \mathrm{reach}_F(T) \qquad \text{and} \qquad \rho(w) \triangleq \max_{F \vdash w} \rho(F)\ .$$

Note that $\rho(w)$ only considers synchronous forks F as that will be our case of interest.

Definition 15 (Relative margin). *For a closed Δ-fork $F \vdash_\Delta w$ we define the* relative margin *of F, denoted $\mu_\ell(F)$, to be the "penultimate" reach taken over tines T_1, T_2 of F such that $T_1 \not\sim_\ell T_2$:*

$$\mu_\ell(F) \triangleq \max_{T_1 \not\sim_\ell T_2} \Big(\min\{\mathrm{reach}_F(T_1), \mathrm{reach}_F(T_2)\} \Big)\ .$$

We again overload the notation by defining

$$\mu_\ell^\Delta(w) \triangleq \max_{\substack{F \vdash_\Delta w \\ F \text{ closed}}} \mu_\ell(F)$$

and write $\mu_\ell(w)$ to denote the synchronous case $\mu_\ell^1(w)$.

Similarly to β_ℓ^Δ in the PoW case, the above quantity has a direct connection to a persistence failure: roughly speaking, if w is a characteristic string capturing the execution of the PoS blockchain up to some current time t, and $\mu_\ell^\Delta(w) < 0$ for some $\ell < t$, then it is guaranteed that the fork $F \vdash_\Delta w$ that resulted from the execution does not allow the adversary to make any honest party at time t adopt a blockchain that differs from its currently held one before (or up to) the index ℓ.

Reduction mapping. We make use of the reduction mapping ρ_Δ introduced in [11] that inspired also the more involved reduction relation for the PoW case introduced in Section 5.3. Note that while keeping notational compatibility with previous work, in ρ_Δ we never drop the subscript Δ, to avoid confusion with the reach mapping ρ of Definition 14.

Definition 16 (Reduction mapping [11]). *For a positive integer Δ we define the function $\rho_\Delta \colon \{0, \mathsf{h}, \mathsf{a}\}^* \to \{\mathsf{h}, \mathsf{a}\}^*$ inductively as follows:*

$$\begin{aligned}
\rho_\Delta(\epsilon) &= \epsilon, \\
\rho_\Delta(0 \,\|\, w) &= \rho_\Delta(w), \\
\rho_\Delta(\mathsf{a} \,\|\, w) &= \mathsf{a} \,\|\, \rho_\Delta(w), \\
\rho_\Delta(\mathsf{h} \,\|\, w) &= \begin{cases} \mathsf{h} \,\|\, \rho_\Delta(w) & \text{if } w \in 0^\Delta \circ \{0, \mathsf{h}, \mathsf{a}\}^*, \\ \mathsf{a} \,\|\, \rho_\Delta(w) & \text{otherwise.} \end{cases}
\end{aligned} \tag{28}$$

We call ρ_Δ the reduction mapping for delay *Δ.*

Note that the above reduction mapping, introduced in [11], translates any honest rounds that are not sufficiently isolated from other honest rounds into adversarial rounds, to keep the accounting on the safe side. It would be possible to adopt a slightly tighter treatment exemplified in the PoW part of our analysis, where such rounds are dropped without incurring a security penalty, however we refrain from that optimization for the sake of compatibility with the analysis in [11].

Again, we also consider a slightly modified version of the reduction mapping, denoted ρ_ℓ^Δ, that takes into account a special position ℓ. It outputs a pair $\rho_\ell^\Delta(w) = (x, y)$, where, intuitively, $\rho_\Delta(w) = xy$ and x is the part of $\rho_\Delta(w)$ that results from processing the first ℓ symbols of w. Formally, $x = \rho_\Delta(w')$ where w' is the longest prefix of w from $(\mathcal{W}_{\mathsf{pos}})^\ell \circ \{0\}^*$ and y is the unique string such that $\rho_\Delta(w) = xy$.

Intuitively, a fork $F \vdash_\Delta w$ can be naturally interpreted as a (synchronous) 1-fork of the reduced string $\rho_\Delta(w)$ by suitably adjusting vertex labels. This is formalized in the following definition and the made formal in Lemma 8, which are implicit in [11].

Definition 17 (Reduction of forks [11]). *Let $w \in (\mathcal{W}_{\mathsf{pos}})^L$, $w' = \rho_\Delta(w)$, and let $F \vdash_\Delta w$. Define F' to be the labeled graph with the same set of vertices and directed edges as F; the labeling function $\mathsf{l}'_\#(\cdot)$ of F' is as follows. Note that $|\rho_\Delta(w)| = |A(w)|$; each non-$0$ symbol of w corresponds to a unique symbol in w'. Let $\pi : A(w) \to \{1, \dots, |A(w)|\}$ be the (bijective, increasing) function which records the position in w' corresponding to a particular index $i \in A(w)$. Then the labeling $\mathsf{l}'_\#(\cdot)$ for F' is given by the rule $\mathsf{l}'_\#(v) = \pi(\mathsf{l}_\#(v))$; $\mathsf{l}'_\#(r) = 0$ for the root vertex r. For convenience, we overload the notation ρ_Δ by defining $\rho_\Delta(F) = F'$.*

Lemma 8 ([11]). *For $w \in \mathcal{W}_{\mathsf{pos}}^L$ and $F \vdash w$, $\rho_\Delta(F) \vdash \rho_\Delta(w)$.*

This has a direct implication for our quantities of interest, the following statement is straightforward.

Lemma 9. *Let $w \in \mathcal{W}_{\mathsf{pos}}^L$ and let $\rho_\ell^\Delta(w) = (x, y)$. Then $\mu_\ell^\Delta(w) \leq \mu_{|x|}(xy)$.*

Proof. Given $w \in \mathcal{W}_{\mathsf{pos}}^L$, consider a closed fork $F \vdash_\Delta w$ and tines $T_1 \not\sim_\ell T_2$ in F such that they witness $\mu_\ell^\Delta(w)$; i.e., $\mu_\ell^\Delta(w) = \min(\mathrm{reach}_F(T_1), \mathrm{reach}_F(T_2))$. Let T_1', T_2' denote the tines corresponding to T_1, T_2 in $\rho_\Delta(F)$, respectively. Observe that for all $i \in \{1, 2\}$, $\mathrm{gap}_F(T_i) \geq \mathrm{gap}_{\rho_\Delta(F)}(T_i')$ as the lengths of all tines are preserved by ρ_Δ and vertices have only changed from honest to adversarial; and similarly $\mathrm{reserve}_F(T_i) \leq \mathrm{reserve}_{\rho_\Delta(F)}(T_i')$. Hence $\mathrm{reach}_F(T_i) \leq \mathrm{reach}_{\rho_\Delta(F)}(T_i)$ for both $i \in \{1, 2\}$ and the statement follows by Lemma 8. □

The synchronous case. The following recursive characterization of $\rho(\cdot)$ and $\mu_\ell(\cdot)$ is given in [23,5].

Lemma 10 ([23,5]). *For any $w \in \{\mathsf{h},\mathsf{a}\}^*$ we have*

$$\begin{aligned}
\rho(\varepsilon) &= 0\\
\rho(w\mathsf{a}) &= \rho(w)+1\\
\rho(w\mathsf{h}) &= \begin{cases} 0 & \text{if } \rho(w)=0,\\ \rho(w)-1 & \text{otherwise.}\end{cases}
\end{aligned}$$

Moreover, if $|w|=\ell$ then $\mu_\ell(w)=\rho(w)$, and for any w' with $w' \geq \ell$ we have

$$\begin{aligned}
\mu_\ell(w'\mathsf{a}) &= \mu_\ell(w')+1\\
\mu_\ell(w'\mathsf{h}) &= \begin{cases} 0 & \text{if } \rho(w') > \mu_\ell(w')=0,\\ \mu_\ell(w')-1 & \text{otherwise.}\end{cases}
\end{aligned}$$

For our investigation we will make use of two direct consequences of the above characterization. First, the quantities ρ and μ_ℓ before position ℓ obey a *barrier* random walk of Def. 12 with a barrier at 0 just like β_ℓ does before ℓ in the PoW case of Lemma 1. Second, after ℓ, μ_ℓ follows a simple biased random walk as long as it has a positive value. This implies that given a string $w \in \{\mathsf{h},\mathsf{a}\}^L$ with $L \geq \ell$, the event of $\mu_\ell(w_1 \dots w_i)$ never hitting 0 in rounds $i \in \{\ell+1,\dots,L\}$ is equivalent to the event that an *absorbing* random walk starting at value $\rho(w_1 \dots w_\ell)$ and determined by the string $w_{\ell+1}\dots w_L$ ends with a positive value.

Gauges and gauge-limited variables. In preparation for our discussion of the characteristic strings that arise in the PoS case, we define the notion of gauges and give a tail bound for variables that satisfy gauging conditions. In the context of a sequence of random variables, a *gauge* is a function which determines a value in the range $[0,1]$ for any particular prefix of values taken by the random variables. We use this for essentially one purpose: to bound the expected value (for a given history of values taken by the random variables) of an associated random variable. In this sense, a gauge is a way to generalize the standard notion of martingale.

Let $\mathcal{A}=(A_1,\dots,A_n)$ be a sequence of random variables, each taking values in a set V. An $\mathcal{A}$-*gauge* is a function $G : V^{<n} \to [0,1] \subset \mathbb{R}$, where $V^{<n} = \{(v_1,\dots,v_k) \mid v_i \in V, 1 \leq k < n\} \cup \{\epsilon\}$; we use ϵ here to denote the empty sequence of variables. A gauge has *limit* L if $\forall (v_1,\dots,v_n) \in V^n$,

$$\sum_{k=0}^{n-1} G(v_1 \dots v_k) \leq L$$

where $(v_1 \dots v_0)$ denotes ϵ, the empty string.

Let $\mathcal{X}=(X_1,\dots,X_n)$ be a sequence of random variables taking values in $\{0,1\}$ so that each X_k is determined by $(A_1,\dots,A_k)$. We say that $\mathcal{X}$ is $(\mathcal{A},G,L)$-*gauge limited* if G is an $\mathcal{A}$-gauge with limit L and for any $1 \leq t \leq n$, $\mathbb{E}[X_t \mid A_1,\dots,A_{t-1}] \leq G$ (note that both of these are functions on V^{t-1}): concretely, for any $1 \leq t \leq n$ and any sequence of values $v_1,\dots,v_{t-1} \in V$,

$$\mathbb{E}[X_t \,|\, A_1=v_1,\dots,A_{t-1}=v_{t-1}] \leq G(v_1,\dots,v_{t-1})\,.$$

When $n \to \infty$ and L is fixed, or slowly growing, gauge limited variables satisfy tail bounds akin to those for Poisson random variables.

Lemma 11. *Let $X_1,\dots,X_n$ be $(\mathcal{A},G,L)$-gauge limited random variables. Then for any $b \geq L$,*

$$\Pr\left[\sum_i X_i \geq b\right] \leq \begin{cases} \exp\left(\frac{-(b-L)^2}{2b}\right), \\ e^{-L}\left(\frac{eL}{b}\right)^b. \end{cases} \tag{29}$$

If, furthermore, the gauge satisfies $G(*) \leq f < 1$ with certainty (in which case the random variables satisfy $\mathbb{E}[X_i] \leq f$ for any setting of $A_1,\dots,A_{i-1}$), then $\sum_i X_i \prec P_\lambda$ for parameter $\lambda = L(-\ln(1-f)/f) = L(1+f/2+f^2/3+\cdots)$.

Proof sketch. A proof by induction establishes that the moment generating function $\mathsf{m}(t)$ of the random variable $\sum_i X_i$ satisfies $\mathsf{m}(t) = \mathbb{E}[e^{t\sum X_i}] \leq e^{L(e^t-1)}$. Recognizing $e^{L(e^t-1)}$ as the moment generating function for the Poisson distribution with parameter L, it follows that any tail bound relying only on the moment generating function for the Poisson distribution (such as those of Fact 1) also applies to gauged random variables. We omit the details.

As for the concluding statement in the setting when $G(*) \leq f < 1$ with certainty, we explicitly construct a coupling between P_λ (a Poisson random variable with parameter $\lambda = L[-\ln(1-f)/f]$) and $\sum X_i$. For convenience, define the "expectation gauge" $G_{\mathbb{E}}(x_1, \ldots, x_{t-1}) = \mathbb{E}[X_t \mid \forall i < t, X_i = x_i]$; it is easy to check that $\mathcal{X}$ is $(\mathcal{X}, G_{\mathbb{E}}, L)$-gauge limited and that $G_{\mathbb{E}}(*)$ is never more than f. For $p \in [0,1]$, let $\Lambda(p)$ satisfy $1 - p = \exp(-\Lambda(p))$; note that if X is a $\{0,1\}$ random variable with $\mathbb{E}[X] = p$, then $\Pr[X = 1] = \Pr[P \geq 1]$ if $P \sim \mathcal{P}_{\Lambda(p)}$. To construct the coupling, consider a Poisson process on the real interval $[0, L(-\ln(1-f)/f)]$ (with the defining property that the number of "arrivals" in any interval of length λ has distribution $\mathcal{P}_\lambda$). Then consider the random variables $Y_i \in \{0,1\}$ given by the following rule: Y_1 is the indicator r.v. for the event that at least one arrival appeared in the interval $[0, \Lambda(G_{\mathbb{E}}(\epsilon))]$. Each subsequent variable Y_t is determined by examining the fresh subinterval

$$\left[\sum_{0\leq i<t-1} \Lambda(G_{\mathbb{E}}(Y_1, \ldots, Y_i)), \sum_{0\leq i<t} \Lambda(G_{\mathbb{E}}(Y_1, \ldots, Y_i))\right].$$

As with Y_1, the variable Y_t indicates an arrival in this new interval. By definition, these have precisely the distribution of the X_i. Noting that $\Lambda(p) = -\ln(1-p)$, we have

$$\Lambda(p) \leq p + p^2/2 + p^3/3 + \cdots \leq p(1 + f/2 + f^2/3 + \cdots) = p(-\ln(1-f)/f)$$

and, in light of the gauge limit L, the process requires length no more than $L(-\ln(1-f)/f)$, as desired. It follows that $\sum_i X_i \prec P_\lambda$ for $\lambda = L(-\ln(1-f)/f)$. □

Distribution of the characteristic string. We now characterize the distribution of the characteristic strings $W_1 \ldots W_n$ that we will be facing in our scenario with temporary adversarial spikes. As is customary in the PoS setting, we will be expressing the stake controlled by parties in a *relative* manner (i.e., as a fraction of the fixed total amount of stake).

In a nutshell, our adversary is allowed to choose in each round k a pair of parameters $(H_k, A_k) \in [0,1]^2$ that determine that in this round, H_k fraction of stake will be alert and A_k fraction of stake will be adversarial, subject to some restrictions. We denote by $B_k \triangleq H_k + A_k \in [0,1]$ the active relative stake in this round. If the choice does not represent a "safe" alert stake ratio $H_k/B_k \geq \alpha$ for some threshold α detailed below, the adversary "pays" for his choice (from his total budget C, accounted in "relative stake" units) the amount $G_k \triangleq H'_k - H_k$ by which H_k would have to be increased to achieve a safe alert stake ratio $H'_k/B_k = \alpha$. The amount of stake G_k is bounded by a W-gauge G with limit C. The adversary is also bound to obey a lower bound on active relative stake, which we denote β in this section, i.e., we assume $B_k \geq \beta \in (0,1]$.

Towards formalizing the distribution of the characteristic string $W_1 \ldots W_n$ that this experiment induces, let us introduce some notation. For any fixed $w = w_1 \ldots w_n$, let $E_{w_{1:k-1}}$ denote the event $W_{1:k-1} = w_{1:k-1}$, and let $z_{w_{1:k-1}} \triangleq \Pr[W_k \neq 0 \,|\, E_{w_{1:k-1}}]$.

Looking ahead, similar to the PoW case, we again need a specific threshold θ_{pos} to bound the ratio of stake of adversarial parties with respect to honest parties. Our analysis is based on the assumption that α satisfies $\alpha(1-f)^{\Delta+1} = (1+\varepsilon)/2$ for some $\varepsilon > 0$, here f is the active rounds coefficient (cf. (27)), which states that the discounted ratio of honest stake dominates the adversarial stake ratio and where ϵ is the gap between the two. Again, the threshold takes the role of this discount factor and we define

$$\theta_{\mathsf{pos}} = \theta_{f,\Delta} \triangleq (1-f)^{\Delta+1}. \tag{30}$$

In round k, conditioned on $E_{w_{1:k-1}}$ we have

$$H_k/B_k \geq (H'_k - G_k)/B_k \geq \alpha - G(w_{1:k-1})/B_k \geq \alpha - G(w_{1:k-1})/z_{w_{1:k-1}} ,$$

as [2] established $z_{w_{1:k-1}} \leq B_k(-\ln(1-f)) \leq B_k$ for $f \leq 1/2$. It was also shown in [2] that this results in

$$\Pr[W_k = \mathsf{h} \,|\, E_{w_{1:k-1}}, W_k \neq 0] \geq (1-f)^2 \cdot (\alpha - G(w_{1:k-1})/z_{w_{1:k-1}}) \geq (1-f)^2\alpha - G(w_{1:k-1})/z_{w_{1:k-1}} \ ;$$

and more broadly, leads to a distribution of W_k governed by the conditions, where $\gamma \triangleq (1-f)^2\alpha$,

$$\begin{aligned} &\Pr[W_k = 0 \,|\, E_{w_{1:k-1}}] \geq (1-f) \\ &\Pr[W_k = \mathsf{h} \,|\, E_{w_{1:k-1}}, W_k \neq 0] \geq \gamma - G(w_{1:k-1})/z_{w_{1:k-1}} \\ &\Pr[W_k = \mathsf{a} \,|\, E_{w_{1:k-1}}, W_k \neq 0] \leq 1 - \gamma + G(w_{1:k-1})/z_{w_{1:k-1}} \end{aligned} \tag{31}$$

for some W-gauge $G(\cdot)$ with limit C; and additionally also

$$\Pr[W_i = 0 \,|\, E_{w_{1:k-1}}] \leq 1 - f\beta \ . \tag{32}$$

We remark that the special case of (31) with $G(\cdot)$ having a limit $C = 0$ was called an (f,γ)-*characteristic condition* in [2].

It is now essential to understand the distribution over (synchronous) characteristic strings induced by ρ_Δ applied to Δ-characteristic strings satisfying (31). In the case of honest majority (i.e., $C = 0$), this was already done in Lemma 12 taken from [2].

Lemma 12 (Structure of the induced distribution [2]). *Let $W = W_1 \cdots W_n$ be a sequence of random variables, each taking values in $\{0, \mathsf{h}, \mathsf{a}\}$, which satisfy (31) with $C = 0$, and let $X = X_1 \cdots X_m = \rho_\Delta(W_1 \cdots W_n)$. Then there is a sequence of random variables $Z_1, Z_2, \ldots$, each taking values in $\{\mathsf{h}, \mathsf{a}\}$, so that*

(i) the random variables $Z_1, \ldots,$ satisfy the $\gamma(1-f)^{\Delta-1}$-martingale conditions of Def. 11;[10]
(ii) $X_1, \ldots, X_{m-\Delta} = \rho_\Delta(W)^{\lceil\Delta}$ is a prefix of $Z_1 Z_2 \cdots$.
(iii) if (32) holds then for any $\delta > 0$ we have $\Pr[m < (1-\delta)f\beta n] \leq \exp\left(-\delta^2 f^2 \beta^2 n/2\right)$.

Hence, $X = \rho_\Delta(W)$ for W according to (31) with $C = 0$ roughly satisfies the martingale condition, and we can make use of the following fact shown in [5].

Lemma 13 ([5]). *Let $X = X_1, \ldots, X_n \in \{\mathsf{h}, \mathsf{a}\}^n$ be a family of random variables satisfying the ξ-martingale condition. Then $\rho(X)$ is stochastically dominated by a random variable R_ξ with distribution $\mathcal{R}_\xi$ defined, for $k \in \{0, 1, 2, \ldots\}$, as*

$$\mathcal{R}_\xi(k) = \Pr\left[R_\xi = k\right] \triangleq \left(\frac{2\xi - 1}{\xi}\right) \cdot \left(\frac{1-\xi}{\xi}\right)^k \ . \tag{33}$$

To account for the effect of the gauge $G(\cdot)$ with a limit $C > 0$ in (31), we will need the following lemma that is a direct consequence of Definition 16.

Lemma 14. *Let $w \in \{0, \mathsf{h}, \mathsf{a}\}^n$ and $x = \rho_\Delta(w)$. Let $\bar{w} \in \{0, \mathsf{h}, \mathsf{a}\}^n$ be a string obtained from w by changing arbitrary d of its "h"-symbols to "a". Then $\bar{x} \triangleq \rho_\Delta(\bar{w})$ satisfies $|x| = |\bar{x}|$, and $\bar{x}$ can be obtained from x by d changes of "h" to "a".*

Putting things together. Similarly to the PoW case, we now study the situation where in an execution of Ouroboros Genesis, an adversarial spike bounded by a limit C occurs before and up to some round ℓ. We upper-bound the probability that after a waiting period D after round ℓ has passed, the quantity μ_ℓ^Δ has not yet reached back 0.

Lemma 15. *Fix some $\alpha, \beta \in (0,1)$, integers $\ell, D > 0$ and denote $L = \ell + D$. Let $W = W_1 \ldots, W_L$ be a family of random variables satisfying both (31) and (32) where*

[10] This assumes that the symbols (h, a) are interpreted as $(0, 1)$ respectively, to match Definition 11. We adopt this interpretation throughout the exposition for simplicity.

(i) $\xi \triangleq \gamma(1-f)^{\Delta-1} = \alpha(1-f)^{\Delta+1} = (1+\varepsilon)/2$ *for some* $\varepsilon > 0$*;*
(ii) G *is a* W*-gauge with limit* C *and with* $G(w_{1:k-1}) = 0$ *for any* $k > \ell$ *and* $w_{1:k-1} \in \mathcal{W}_{\mathsf{pos}}^{k-1}$.

Then there exists some $D_0 = O(fC)$ *such that if* $D > D_0$ *then we have*

$$\Pr\left[\begin{matrix}\forall i \in \{\ell+1,\ldots,L\}\colon \\ \mu_\ell^\Delta(W_{1:i}) > 0\end{matrix}\right] \leq \exp(-\Omega(D)) \; .$$

Proof sketch. Let $\rho_\ell^\Delta(W) = (X,Y)$, $\ell' \triangleq |X|$ and $L' \triangleq |XY|$. We will investigate $\mu_{\ell'}(XY_{1:i})$, as Lemma 9 gives us

$$\Pr\left[\begin{matrix}\forall i \in \{\ell+1,\ldots,L\} \\ \mu_\ell^\Delta(W_{1:i}) > 0\end{matrix}\right] \leq \Pr\left[\begin{matrix}\forall i \in \{1,\ldots,L'-\ell'\} \\ \mu_{\ell'}(XY_{1:i}) > 0\end{matrix}\right] .$$

We consider a sequence of random variables $\widehat{W} = \widehat{W}_1 \ldots \widehat{W}_L$ coupled with W, such that $W_i \in \{0,\mathsf{h}\}$ implies $\widehat{W_i} = W_i$, and for every $w_1 \ldots w_n \in \mathcal{W}_{\mathsf{pos}}^n$ and $k \in [L]$,

$$\Pr[\widehat{W_k} = \mathsf{h} \mid E_{w_{1:k-1}}, W_k = \mathsf{a}] = \min\left\{\frac{G(w_{1:k-1})}{\Pr[W_k = \mathsf{a} \mid E_{w_{1:k-1}}]}, 1\right\} \tag{34}$$

and $\widehat{W_k} = \mathsf{a}$ otherwise. Denote $(\widehat{X},\widehat{Y}) = \rho_\ell^\Delta(\widehat{W})$.

Note that $\widehat{W}$ satisfies (31) with $C = 0$ and by Lemma 12, the distribution of the random variable $\widehat{X}\widehat{Y}$ satisfies the $\gamma(1-f)^{\Delta-1}$-martingale condition (except for the constant-length Δ-suffix). Therefore, by Lemma 13, $\mu_{\ell'}(\widehat{X}) = \rho(\widehat{X})$ is stochastically dominated by R_ξ for $\xi = \gamma(1-f)^{\Delta-1}$ and hence for any t,

$$\Pr[\mu_{\ell'}(\widehat{X}) \geq t] = \exp(-\Omega(t)) \; .$$

Moving to $\mu_{\ell'}(X)$, one can easily verify from (31) and (34) that the sequence of indicator random variables $\{W_k \neq \widehat{W}_k\}_{k=1}^L$ is (W,G',C')-gauge limited. For $G'(\bar{w}) \triangleq f \cdot G(\bar{w})$ for each $\bar{w} \in \mathcal{W}_{\mathsf{pos}}^{<L}$, and $C' \triangleq f \cdot C$. Hence by Lemma 11 for any $t \geq 0$,

$$\Pr\left[\left|\left\{i \colon W_i \neq \widehat{W_i}\right\}\right| \geq fC + t\right] \leq \exp\left(fC - \Omega(t)\right) \; . \tag{35}$$

By Lemma 14, equation (35) also bounds the number of positions in which X and $\widehat{X}$ differ. Thanks to the exchange principle of Lemma 3, we can shift the effect of these additional "a"-symbols to the end of X without decreasing $\rho(X) = \mu_{\ell'}(X)$, which performs a barrier walk by Lemma 10. Again by Lemma 10, each such "a"-symbol at the end simply increases the quantity $\mu_{\ell'}(\cdot)$ by 1. We can hence conclude that for any $t > 0$,

$$\Pr\left[\mu_{\ell'}(X) > fC + t\right] \leq \exp(fC - \Omega(t)) \; . \tag{36}$$

For $i \geq 1$, the value $\mu_{\ell'}(XY_{1:i})$, while positive, follows a simple negatively-biased random walk by Lemmas 10 and 12 (except for the short Δ-suffix that affects the outcome of the walk by at most Δ). Hence the probability that it remains positive for all $i \in [|Y|]$ is upper-bounded by the probability that $\mu_{\ell'}(XY) > 0$. By Lemma 12(iii), the length of this walk $|Y|$ is at least $f\beta D/2$ except with error at most $\exp(-f^2\beta^2 D/8)$ and hence

$$\Pr\left[\mu_{\ell'}(X) - \mu_{\ell'}(XY) < \frac{\gamma(1-f)^\Delta f\beta D}{4}\right] \leq \exp(-\Omega(D)) \tag{37}$$

via the standard Azuma inequality for martingales. Combining equations (36) and (37) again concludes the proof. □

Analogously to the PoW section, we can summarize the main result as follows:

Theorem 2. *The Nakamoto-style PoS blockchain protocol in the single-epoch setting, executed over a network with maximum delay Δ, active rounds coefficient f and cut-off parameter κ satisfies the consistency self-healing property against any $(\theta_{f,\Delta}, \epsilon, \beta, \mathcal{B})$-adversary, where the threshold is defined in equation 30, and where $\epsilon, \beta > 0$ are arbitrary constants, with a vulnerability period given by the pair (τ_l, τ_h) where $\tau_l = O(\mathcal{B})$ and $\tau_h = O(\mathcal{B}) + O(\kappa)$.*

Proof sketch. The reasoning is analogous to the proof of Theorem 1, focusing on the quantity μ_ℓ^Δ instead of β_ℓ^Δ and invoking Lemma 15 instead of Lemma 7. □

6.4 Explicit Bounds on Recovery from Adversarial Majority

For our experimental results, we employ the same sequence of reasoning as was used for proving Lemma 15. However, instead of an asymptotic treatment, we plug in concrete bounds for all involved distributions. More concretely, we numerically simulate a particular negatively biased random walk with an absorption barrier at 0, parametrized as follows. The initial value of the random walk is sampled from a distribution that is a convolution of the distribution $\mathbb{R}_\xi$ from (33) upper-bounding $\mu_\ell(\widehat{X})$ and a distribution bounding from above the effect of a spike as in (29). In this concrete setting, we use the stronger tail bounds of Lemma 11 and the observation that if X is a random variable on $\{0, 1, \ldots\}$ satisfying a tail bound $T(k)$ (which is to say that $\Pr[X \geq k] \leq T(k)$), then X is stochastically dominated by the distribution $k \mapsto T(k) - T(k+1)$. This gives a concrete, stochastically dominant distribution for the simulations.

We select $f = 1/20$, $\Delta = 3$, and $\gamma = .9$ (the lower bound on the stake ratio of alert to active parties). Then $\gamma(1-f)^\Delta = .7716$ is the effective bias of the resulting walk after reduction. As the random walks correspond to the random variables *after* reduction, we scale the time axis by $1/f = 20$ for a fair comparison against actual protocol time. As argued right after Lemma 10, we can simulate an absorbing random walk since the probability that this walk remains positive in fact upper-bounds the probability that the simple random walk performed by $\mu_{\ell'}$ between rounds ℓ' and L' never hits zero. Our results are summarized in Figures 5, 6, and 7.

The spike quota can again be seen as the expected number of "extra" successes by the adversary in the slot-leadership process. More precisely, s directly corresponds to the gauge limit established in the proof of Lemma 15 to bound the "extra" successes of the adversary (cf. equation 35). The distribution of those extra successes is by Lemma 11 dominated by a Poisson distribution with parameter $\lambda = s(-\ln(1-f)/f)$ which yields a nice interpretation of the role of the quota s acting in the role of the gauge limit L in Lemma 11.

6.5 Adversarial Spikes: Multiple Epochs and Genesis Rule

The single-epoch analysis comes with a few amenities. For example, since we never adjust to the stake-distribution, the negative effects of long-range attacks are not considered, and the Ouroboros protocol can be seen as a longest-chain-rule Nakamoto consensus mechanism [11,2]. Lifting to multiple epochs requires additional adjustments as discussed in Section 6.1, we briefly recall them here. Each epoch is a sequence of R rounds. At the start of an epoch, say at round t, the snapshot of the stake-distribution is taken from the block up to round $< t - g$, where $g = R$ in [11,2], and the new epoch's randomness is seeded by blocks up to round $< t - h$, again where $h = R/3$. This gives rise to the epoch structure depicted in Figure 8 consisting of three important intervals called *phases*: the first $O(\kappa)$ rounds ensure stabilization of the stake distribution before the second phase via a combination of the chain-growth (CG) and common-prefix (CP) properties. The purpose of the second phase is to guarantee an honest block in this interval in any chain stabilized at the end of the third phase (which again takes $O(\kappa)$ rounds). The size of the second phase is derived from the existential chain quality ($\exists\mathsf{CQ}$) guarantee of Nakamoto consensus and in general is only a fraction of the size of the first (and third) phase. Taking all that into account it follows $R = O(\kappa)$.

In terms of security, the following properties are relevant: if $B_{<t-h}$ denotes the last block that affects the randomness, then at the start of the new epoch, $B_{<t-h}$ must be part of the immutable ledger state. Likewise, at the start of the second phase of the epoch, the last block $B_{<t-g}$ of the previous epoch that fixes a new stake distribution must be part of the immutable ledger state. Finally, at the end of the epoch, parties must

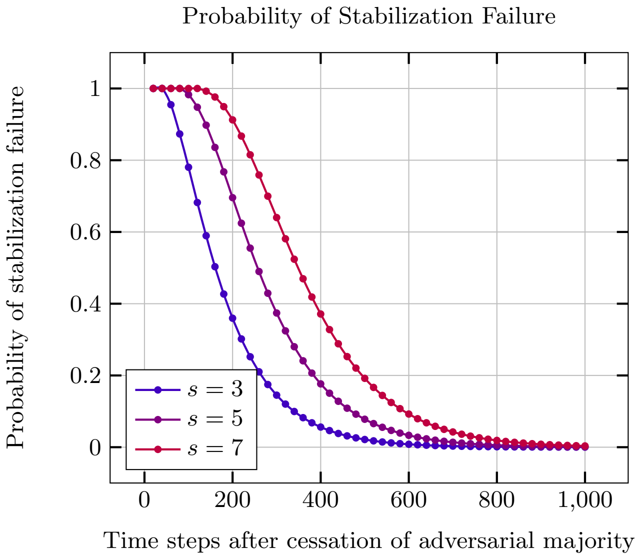

Fig. 5. Graphs of the probability that a PoS blockchain fails to stabilize after a period of adversarial majority. These plots correspond to a 10% adversary, $\varDelta = 3$, and $f = 1/20$; adversarial quotas of $s = 3$, 5, and 7 are shown.

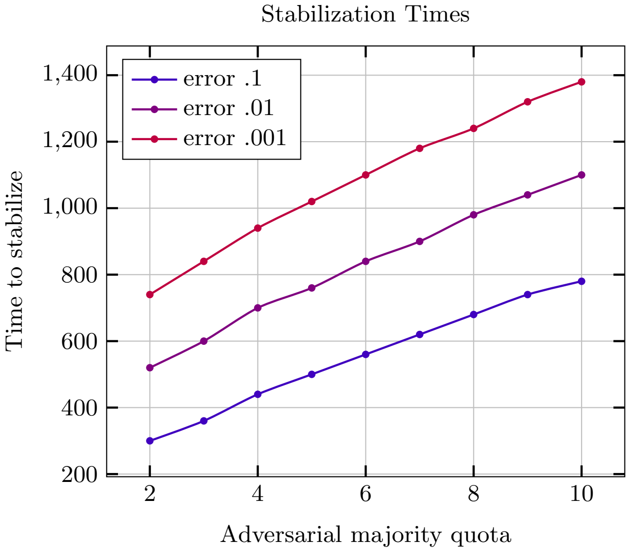

Fig. 6. Graphs of the time required to stabilize with error no more than 1/10, 1/100, and 1/1000. As above, this assumes a 10% adversary, $\varDelta = 3$, $f = 1/20$; quotas are given by the x-axis.

again have reached agreement on the next epoch randomness, and the randomness must be affected by at least one honest block minted in the second phase of the epoch.

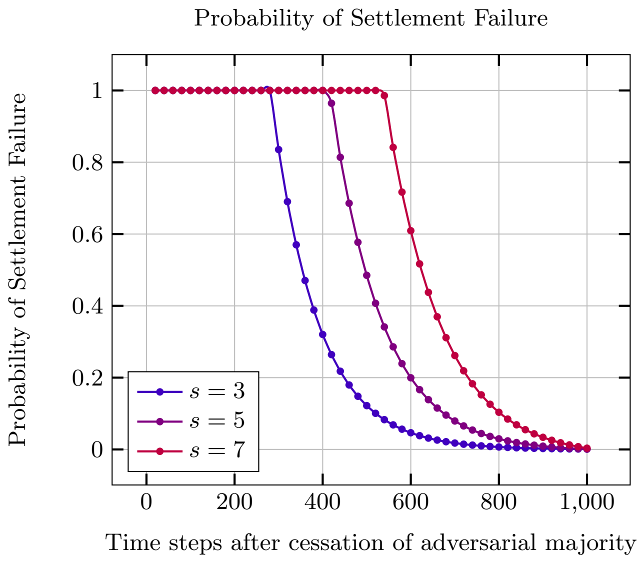

Fig. 7. Graphs of the probability that a PoS blockchain fails settlement after a period of adversarial majority. These plots correspond to a 10% adversary, $\Delta = 3$, $f = 1/20$, and adversarial quotas of $s = 3$, 5, and 7.

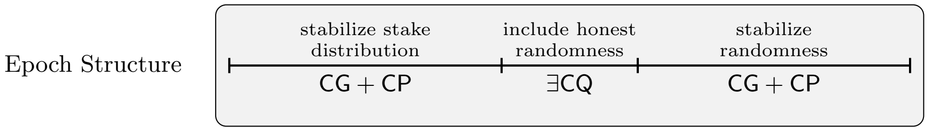

Fig. 8. Illustration of the three different intervals of an epoch for the inductive argument underlying Genesis.

Ouroboros Genesis can be parametrized to handle a given anticipated spike profile with budget $\mathcal{B}$ that might be taken as an additional parameter or itself be expressed as a, for example, linear function of a security parameter (which is often equated with the above parameter κ). Let us assume for the sake of the argument that different intervals of adversarial dominance are separated (time-wise distant enough) be treated and foiled individually (if too close, we would have to model them as one connected interval). Increasing the length of each phase of an epoch appropriately as a function of a spike is sufficient to contain a spike profile. For example, a spike budget $\mathcal{B}$ linear in κ would amount to a linear expansion (in κ) to achieve a sufficiently small failure probability. By Theorem 2, we conclude that damage to the ledger state is limited and self-healing is possible thus not preventing stabilization in the first and third pahses, and leaving enough unaffected rounds to invoke the liveness guarantees of Nakamoto consensus of Section 7 for the second phase. The resulting epoch length would still satisfy $R = O(\kappa)$.

Finally, the Genesis rule adds one more restriction to the above equations, namely that spikes of adversarial dominance are tolerated as long as they are substantially shorter than the Genesis window s (which in an asymptotic treatment is linear in the security parameter), such that over a sequence of s slots the effect of the spike is fully contained. This is tight for Genesis: given the defined joining procedure for new parties, it is clear that any interval of adversarial majority of length at least s rounds leads to an s-round interval I such that the adversary can at any later point forge a chain that forks right before the beginning of I, and is more dense on I than the honestly created chain, hence dominating the honest chain according to the Genesis rule. This can be exploited in a straightforward way to violate persistence.[11]

7 Liveness in PoW and PoS Blockchains

Consider a characteristic string $w \in \mathcal{W}_{\mathsf{pow}}^{L}$; we write $w = ((h_1, a_1), \ldots, (h_L, a_L))$. To study liveness, we define some simple quantities derived from w. For each round r we define $I_r \in \{0, 1\}$ so that $I_r = 1$ if r is uniquely honest and left Δ-isolated. (That is $I_r = 1$ if and only if $1 = h_r = \sum_{i=r-\Delta}^{r} h_i$, where we define $h_i = 0$ for all $i \leq 0$ for this purpose.) Then, for each round r and $d \geq 0$, we define

$$C_{r,d} = \begin{cases} 1 & \text{if } \sum_{i=1}^{d-\Delta} I_{r+i} > \sum_{i=0}^{d} a_{r+i}, \\ 0 & \text{otherwise.} \end{cases}$$

When $C_{r,d} = 1$, we say that round r is *d-covered.* Then define

$$L_s^{(u)}(w) = \bigwedge_{r \leq s-u} C_{r,s-r}\,.$$

(Of course, I_r and $C_{r,s}$ are functions of w, as is $L_s^{(u)}$, but we suppress this in the notation.) Our principal analytic tool for establishing liveness is the following, which applies to both the PoW and PoS settings.

Lemma 16. *Any execution, in the PoW or PoS settings, associated with characteristic string w achieves liveness u if $L_s^{(u)}(w) = 1$ for all rounds s.*

Proof. Any blockchain broadcast during a PoW or PoS protocol contains a "last honest block" (as we treat the genesis block as honest). One perspective on liveness is to ensure that this last honest block cannot occur too early. Consider then the view of a participant in such a protocol at round s and a particular, prior round r. Observe that if there have been more uniquely-honest, left Δ-isolated rounds than total adversarial blocks between rounds r and s, then no honest block generated in round r can possibly be the last block on any longest chain observed in round s, for the simple reason that the longest chain must have grown by at least the number of left-isolated rounds (and there are insufficient adversarial blocks to build a competing extension). Some care is required in this argument to ensure that the left-isolated rounds in question are at least Δ rounds prior to s (to ensure that the chain will have been delivered to any observer in round s). To summarize: an honest block generated in round r can be the last honest block on a longest chain observed in round s only if $C_{r,s-r} = 0$. To ensure liveness (at round s)—that is, that any longest chain observed in round s contains an honest block among the previous u rounds—it thus suffices for $L_s^{(u)}(w) = 1$. □

To assess the effect of adversarial spikes on liveness, first recall that our notion of liveness captures the property of honest blocks being included into the ledger state as we make progress over time. If the onset of a spike is round ρ_α, then any liveness guarantee, say with parameter e, established by the protocol in a setting without spikes, holds for all rounds up to and including round $r < \rho_\alpha - u$. Furthermore, as we know from Theorem 1 and Theorem 2, given the adversary's spike profile some part of the history is guaranteed to stay unaffected. Therefore, it remains to prove that the protocol will return to normalcy after the spike, i.e., that there is a point in time after which the ledger state is guaranteed to be extended by honest blocks with the same guarantee as in a spike-free setting.

Thus we focus on rounds after the conclusion of a spike and consider the event $E(\theta, \epsilon, n_0, \mathcal{B}, \rho_\beta; s, u)$ that $L_s^{(u)} = 0$ with a $(\theta, \epsilon, n_0, \mathcal{B})$-adversary with spike terminating prior to round ρ_β. Writing $D = s - \rho_\beta$, observe

[11] Note that if it was acceptable to assume a trusted "checkpointing service" that serves to every late joiner a valid up-to-date state of the blockchain, as is assumed by protocols preceding Genesis such as [23,10,11], there would be no need for the Genesis rule and the above restriction would not arise.

that

$$L_s^{(u)} = \underbrace{\left(\bigwedge_{s-D \leq r \leq s-u} C_{r,s-r} \right)}_{(\dagger)} \wedge \underbrace{\left(\bigwedge_{r \leq s-D} C_{r,s-r} \right)}_{(\ddagger)} .$$

As the adversarial spike cannot affect rounds after $s - D$, we note that the event "$(\dagger) = 0$" can be bounded above by the probability of a liveness failure in a protocol with no spike. In particular, we note:

$$\Pr[E(\theta, \epsilon, \mathcal{B}, \rho_\beta; s, u)] \leq \Pr[E(\theta, \epsilon, 0, 0; s, u)] + \Pr[(\ddagger) = 0] ,$$

where the first term is the probability of such a liveness violation in the "spike-free" setting, and the second probability is taken with respect to a $(\theta, \epsilon, n_0, \mathcal{B})$-adversary. Thus it suffices to bound above $\Pr[(\ddagger) = 0]$. Our goal is to show that for rounds sufficiently far ahead of ρ_β, we have returned to normalcy. Thus, we write

$$\Pr[(\ddagger) = 0] \leq \max_{s \geq \rho_\beta + D_0} \Pr \left[\bigvee_{r \leq \rho_\beta} \overline{C_{r,s-r}} \right] \triangleq \mathsf{LI}_{D_0} ,$$

now using D_0 to reflect a lower bound on the gap between s and ρ_β; the probability is taken with respect to a $(\theta, \epsilon, n_0, \mathcal{B})$-adversary.

Theorem 3. *The Nakamoto-style PoS and PoW blockchain protocols satisfy the liveness self-healing property with parameter u against a $(\theta, \epsilon, n_0, \mathcal{B})$-adversary for $\theta \leq \theta_{\mathsf{pos}}, \theta_{\mathsf{pow}}$, respectively, and $\epsilon, n_0 > 0$ with the vulnerability period given by the pair $(\tau_l = u, \tau_h)$ where $\tau_h = O(\mathcal{B}) + O(u)$.*

Proof. We are somewhat more brief, as the conclusions follow directly from the techniques developed in Sections 5 and 6 above. The event $\mathsf{LiveFail}_u(r)$ is unaffected by the adversarial spike if $r + u < \rho_\alpha$, which determines the value of τ_l in the statement of the theorem. As for $\mathsf{LiveFail}_u(r)$ when $r > \rho_\beta$, by the discussion above it suffices to appropriately select τ_h to ensure that LI_{D_0} is negligible (in u) for all $D_0 \geq u + \tau_h$. Consider an arbitrary event $C_{r,s-r}$ for which $s - r = D$—in general, the region between r and s may intersect (or follow) the region defining the spike. In the PoW case, we follow the proof of Lemma 7 which controls precisely this event: combining Equation (24), which bounds the number of "a" symbols arising from the adversarial spike, and Equation (25), which bounds the gap between "a" symbols (unrelated to the spike) and (doubly) isolated "h" symbols, we find that there is a $D_0 = O(\mathcal{B})$ so that for any $D \geq D_0$, $\Pr[\overline{C_{r,s-r}}] \leq \exp(-\Omega(D))$. Then, we find that

$$\mathsf{LI}_{D_0} \leq \sum_{D \geq D_0} \exp(-\Omega(D)) = \exp(-\Omega(D_0)) .$$

Thus, we may select $\tau_r = O(\mathcal{B} + u)$ so that $\mathsf{LI}_{D_0} \leq \exp(-\Omega(u))$ for all $D_0 \geq \tau_h$, as desired. The PoS case is similar, borrowing the analogous tail bounds from Lemma 15. □

8 Iterated BFT Protocols

The third category of blockchain protocols are iterated BFT (Byzantine fault tolerant) protocols. These include tendermint, hotstuff, algorand and others [27,40,9]. The main structure of such protocols entails the (not necessarily black-box) self-composition of an underlying BFT protocol Π_{BFT}. In this section, we show how our theoretical model of a ledger abstraction is rich enough to also be able to capture this type of protocol. For completeness, we then illustrate how the security of Algorand can be obtained in this formalism and how one can investigate Algorand against adversarial spikes, setting the stage for future analyses of Algorand dealing with recovering from such adverse settings which is declared as a future extension to the protocol in [9]. Recall that one reason to introduce our model is to provide a unifying language for ledger protocols (which is beneficial for relating concepts and security claims) which at the same time is less complex than the UC model.

In iterated BFT, protocol execution in the view of any honest party is comprised of a number of iterations of an underlying BFT protocol, say Π_{BFT}, which operates with respect to a stakeholder distribution $\mathbb{S}_i$. Without loss of generality the time is divided in slots assuming a synchronous mode of execution. In all existing protocols it holds that there is a polynomial-time predicate P for which it holds that $P(i, j, st, sk)$ is true if and only at the i-th slot, the party in possession of the j-th coin, is supposed to act given the protocol is at state st and the party's private key is sk. Depending on the protocol there maybe more than one such predicate. For instance in Algorand [9] we have a predicate $P()$ that determines a "proposer" and another predicate $V()$ that determines a "verifier"; proposers are organizing transactions while verifiers ratify them. These predicates are publicly verifiable.

To be more concrete, we focus on the Algorand protocol [9]. The protocol proceeds in block-production phases, where each phase entails a number of steps: Step 1 is the block proposal stage, where each potential leader creates and disseminates a block. Step 2 and 3 comprise the graded consensus (GC) subprotocol. Each elected verifier chooses its input to this subprotocol to be the valid block received from the proposer with the smallest derived index[12] Depending on whether a block receives enough support by means of at least t signed confirmation messages of honest verifiers (in both steps of GC), where t is some threshold, the graded consensus will end with each party holding a decision which block to append. In case the support is high, an honest verifier concludes the GC step with a high grade (and has some block B) and sets an indicator bit b to 0. If support is too low, a party may hold a block B or the empty block that it would append to the chain, and defines the indicator b to be 1. With this, the parties finally enter the binary byzantine agreement (BBA*) stage, at step 4, to reach consensus on the indicator bit b. Roughly speaking, if the agreement reached is $b = 0$ (strong support), then the corresponding block that received strong support is output. In the other case, an empty block is produced. Note that the protocol's security is based on Byzantine quorums of the involved verifiers in each step and the protocol is proven secure and live as long as the set of honest verifiers h_i (in each step i) has size at least t, and, if $h_i + 2a_i < 2\mathsf{t}$, where a_i denotes the set of malicious verifiers (in each step i).

Instantiating the model for iBFT We provide an abstract framework to capture the relevant events in Algorand. As seen above, we have two types of rounds. The rounds have some common events. We assume one coin per identity (and perfect synchrony for simplicity). We now describe the elements $(\mathbb{P}_{\mathsf{ibft}}, \Sigma_{\mathsf{ibft}}, \mathcal{W}_{\mathsf{ibft}}, \mathcal{D}_{\mathsf{ibft}}, \mathcal{G}_{\mathsf{ibft}}, \dot{\to}_{\mathsf{ibft}}, \mathbb{L}_{\mathsf{ibft}})$ of the ledger abstraction. Each round has a number of honest and adversarial entities, again with an arbitrary identity set $\mathbb{P}_{\mathsf{ibft}}$ and $\Sigma_{\mathsf{ibft}} \subseteq \mathbb{N} \times \mathbb{N}$ to denote the assignment of coins to honest and adversarial entities, respectively. In each round three relevant items are sampled: the numbers h_i and a_i of honest and adversarial committee members, and an indicator $I \in \{\mathsf{hwin}, \mathsf{awin}\}$ describing whether an honest or adversarial committee member has the smallest hash. In Algorand, the smallest hash can be bound to a particular party because the value hashed is a (fresh) signature. In the random oracle model where hashes are idealized, the distribution $\mathcal{D}_{\mathsf{ibft}}$ is a straightforward random sampling of the committee identities (the probability of being proposer equals parameter p_l and the one of being a verifier equals parameter p_v) as well as the ordering of the relevant hashes (modeled as uniform values). Since proposer and validation rounds are different in the role they take in the protocol, we define two characteristic strings, w^{p} and w^{v} (independent and identically distributed) each defining a sequence of values $w_i = (h_i, a_i, I)$ (which is an equivalent way to express that $\mathcal{W}_{\mathsf{ibft}} = (\mathbb{N} \times \mathbb{N} \times \{\mathsf{hwin}, \mathsf{awin}\})^2$) and we can define the iBFT execution graph.

Definition 18. *A* iBFT execution graph *for the pair of characteristic strings $w^{\mathsf{p}}, w^{\mathsf{v}}$ is a directed, rooted tree $F = (V, E)$ together with a function $\mathsf{l}_{\mathsf{meta}} : V \to \{\mathsf{p}, \mathsf{v}\} \times \{\mathsf{a}, \mathsf{h}\} \times \mathcal{W}_{\mathsf{ibft}} \times 2^{\mathcal{P}}$ and the function $\mathsf{l}_{\#}$ that simply assigns the depth to each node. The following two properties must be satisfied:*

(i) (Well-defined characteristic string) All nodes with the same depth in the tree are given the same symbol of $\mathcal{W}_{\mathsf{ibft}}$ in their $\mathsf{l}_{\mathsf{meta}}$ label.

(ii) (Every vertex corresponds to a protocol step) Each vertex is assigned a label and the vertices at depth one are marked as proposer vertices.

[12] The index is the the hash of the credentials of the proposer, where the credential is a round and seed-dependent signature.

We stress that the above only states the minimum to be a valid abstraction of an execution: it says that the depth of the tree corresponds to a particular round of the algorithm and that the very first round starts with a proposal and that all parties agree with this. Note that we put this auxiliary information of the role of a node, i.e., proposer or verifier, right at the beginning of the label for better readability (whether a node stands for honest/adversarial proposal or honest/adversarial verification round) and define the transition induced by the Algorand protocol next:

Definition 19. *Defines the set $\{G' | G \overset{w_i}{\to} G'\}$, where G satisfies Definition 18 and furthermore, assume G specifies the correct elements of $w_i = (w_i^{\mathsf{p}}, w_i^{\mathsf{v}})$ in its vertices, i.e., at depth $i \geq 2$, the label specifies w_i.*

(i) *if $w_i = (h_i, a_i, \cdot, \cdot)$, $h_i > \mathsf{t}$ there is at least one vertex in G' at depth i extending a tine of G. This happens when there are enough honest parties to validate a voting proposal.*
(ii) *if $w_i = (h_i, a_i)$, $\frac{1}{2}h_i + a_i \leq \mathsf{t}$ then there is, per tine of G, at most one vertex extending the tine to depth i. In addition, any new node at depth i must have type $(\cdot, \mathsf{h}, \cdot, \cdot)$. This suggests a setting where verification is unambiguous.*
(iii) *On each tine, a proposer vertex (i.e., type $(\mathsf{p}, \cdot, \cdot, \cdot)$) is followed by at least four validation-vertices, i.e., of type $(\mathsf{v}, \cdot, \cdot, \cdot)$.*
(iv) *(Phase Ending Condition 1) If a sequence of vertices starting at depth i and ending at depth $i+4$ on a tine are labeled $(\mathsf{p}, \mathsf{h}, \cdot, \cdot), (\mathsf{v}, \mathsf{h}, \cdot, \cdot), \cdot (\mathsf{v}, \mathsf{h}, \cdot, \cdot)$, then the vertex at depth $i+5$ on this tine is labeled $(\mathsf{p}, \mathsf{h}, w_i, \cdot)$ if $w_i^{\mathsf{p}} = (\cdot, \cdot, \mathsf{hwin})$ and labeled $(\mathsf{p}, \mathsf{a}, w_i, \cdot)$ in case $w_i^{\mathsf{p}} = (\cdot, \cdot, \mathsf{awin})$.*
(v) *(Phase Ending Condition 2), On any tine, if we have a vertex $(\mathsf{p}, \mathsf{a}, \cdot, \cdot)$ at depth i followed by continuous sequence of validator vertices $(\mathsf{v}, \mathsf{h}, \cdot, \cdot)$ where $i' \geq i + s$, with $s \geq 7, s - 2 \equiv_3 2$, denotes the smallest index s.t. $(\mathsf{v}, \mathsf{h}, w_i, \cdot)$, then if the vertex at depth $i'+1$ has label $(\mathsf{v}, \mathsf{h}, \cdot, \cdot)$ then $i'+2$ is a proposer vertex restricted by w_i as above.*
(vi) *A vertex can be labelled by $P \in \mathbb{P}$, only if it is a terminal vertex in G.*
(vii) *If $P \in \mathbb{P}$ labels a vertex v in G' then P can only label a vertex v' in G' that is a descendant of v.*

From an execution graph, the extraction algorithm $\mathbb{L}$ to extract a blockchain (per party) works in the straightforward way: From the execution graph G, we get the chain structure (i.e., the minor of G) $\mathbb{L}^P(G)$ of party P by mapping the tine labelled by P to a sequence of blocks by ignoring v vertices and keeping the p vertices except the last one (and connecting them by directed edges in the order appearing on the tine).

We argue below that the above abstraction is an accurate representation of the Algorand execution and hence all the properties, in particular consistency and liveness must hold under the typical assumptions of full participation and a honest super-majority as specified in [9]. Finally, the model allows the protocol to be looked at in adverse circumstances, where the actual fraction of honest active participants is temporarily reduced in the committees below the minimally required two-thirds ratio.

Theorem 4. *Under the assumptions of [9, Theorem 1], that is full participation and (an above two-thirds) super-majority of honest coins n that corresponds to a security threshold $\theta_{\mathsf{ibft}} \triangleq 1/2$ and some gap $\epsilon > 0$), with overwhelming probability over the randomness in the settlement experiment instantiated for the iBFT setting as above, the execution graph structure is a simple chain and any chain structure of any honest party contains a certain fraction p_h of vertices with label h in expectation.*

If the adversary belongs to the class of $(\theta_{\mathsf{ibft}}, \epsilon > 0, n_0 := n, \mathcal{B} := 3(1+\epsilon')(\mathsf{t} \cdot \delta + 1)/p_v)$-adversaries, for any constants $\epsilon' > 0$ and $\delta > \epsilon$, then the vulnerability period is characterized by $\tau_l = 0$ and $\tau_h = \infty$, that is, for any given number r, there is an adversary in the above experiment that generates with substantial probability an execution graph where the vulnerability period is lower bounded by $\tau_h > r$.

Proof sketch. To see that the above chains can indeed be obtained by the protocol run, we distinguish two cases: for the first part, we can invoke the existing analysis of Algorand. Assuming the corruption bound is satisfied, then the only admissible structure is a path and hence a single blockchain that does never split. This is the case because Properties (ii) and *(ii)* of Definition 19 are fulfilled exactly the consistency properties proven by [9, Theorem 1]. Regarding the sequence of block proposer and validation rounds, if the block proposer is honest, then no more than four validation rounds are needed (which is the minimum by definition of the protocol) as given by [9, Theorem 1, property 2] justifying properties (iii) and (iv);

and finally, if the block proposer is adversarial, then the Algorand ensures termination with overwhelming probability by [9, Theorem 1, property 3] as it ensures conclusion of a sequence of validator rounds whenever a common-coin round is won by an honest validator (in which case no more than two more rounds are needed to conclude) which justifies property (v) (note that the condition on s in property (v) reflects a particular round structure of Algorand). Having established this connection, the guarantees of [9, Theorem 1] directly apply, where consistency is obvious and for liveness, [9, Theorem 1, property 4] establishes that the probability p_h of an honest proposer vertex is lower bounded by $h^2(1+h-h^2)$, where h is the assumed lower bound on the total fraction of coins owned by the honest parties in the execution.

If we leave the 1/3-corruption bound then we observe that a fork can emerge. We briefly argue how an adversary can indeed achieve such a structure with substantial probability in case the corruption threshold is violated for 3 steps in a row (hence the factor 3 in the budget $\mathcal{B}$). Let a_i and h_i denote the number of corrupted and honest verifiers in each step and assume the block proposer is corrupted: the adversary proposes two blocks B_1 and B_2 and schedules the delivery such half of the potential verifiers would accept B_1 as input to GC and the other set would accept B_2 as input to GC. Therefore, in GC, the actual verifier sets sign different values with equal expected support. To ensure both blocks have enough support above threshold t for the next step, the adversary needs to corrupt enough participants to ensure that $1/2h_i+a_i \geq \mathsf{t}$ in this round. Next, the adversary schedules the messages as before to achieve an expected equal sized set of verifiers where one set supports B_1 and the other B_2. In order for the final step of GC to conclude with high grade for all verifiers, we again need that $1/2h_i + a_i \geq \mathsf{t}$. The adversary has thereby created a situation where honest parties are split in two sets, all have high grades (but each set for a different block). Honest parties will therefore all terminate in BBA* given that the adversary can fill up with enough signatures which is possible when $1/2h_i + a_i \geq \mathsf{t}$ holds in this step. If the adversary succeeds, the protocol does not try to do a fork resolution[13], and even if the adversary does play honestly (and respecting the corruption bound again) in the following rounds the forking situation remains indefinitely which yields the first part of the second statement. Note that an extension of the attack, in which the adversary is passive after the above active attack but keeping the stake distribution unchanged (and hence still violating the corruption bound), would result in larger diverging paths growing both with empty blocks. For the last part of the statement, note that no attack can revert the already finalized state due to the instant confirmation under adaptive security even in the adverse setting we look at, which yields $\tau_l = 0$ and concludes the study of the vulnerability period. □

Remark 2. It is worth noting the contrast between Theorem 4 and Theorems 1, 2. Recall that in these theorems, the budget $\mathcal{B}$ can be arbitrarily large without invalidating the self-healing property of the protocol. Moreover, in the case of a PoS with multiple epochs of length R, where in each epoch the stakeholder distribution is adjusted, self-healing holds as long as the budget $\mathcal{B}$ is a fraction of the epoch length R, cf. Section 6.5. In contrast to those results, Theorem 4 above suggests that even for a small adversarial budget $\mathcal{B}$ (it holds that $\mathcal{B}/n_0 \leq \mathcal{B}/\mathsf{t} = O(1)$), the protocol fails to self-heal indefinitely. To put it differently, in the iBFT setting, there is no protocol parameter that may be calibrated to accomodate a given upper bound on adversarial spike tolerance. On the other hand, this flexibility in Nakamoto-style PoS comes at the expense of outdating the stakeholder distribution over which the honest majority assumption applies, with the temporal penalty being linear in the level of adversarial budget tolerated during the spike.

References

1. G. Avarikioti, L. Käppeli, Y. Wang, and R. Wattenhofer. Bitcoin security under temporary dishonest majority. In I. Goldberg and T. Moore, editors, *FC 2019*, volume 11598 of *LNCS*, pages 466–483. Springer, Heidelberg, Feb. 2019.
2. C. Badertscher, P. Gazi, A. Kiayias, A. Russell, and V. Zikas. Ouroboros genesis: Composable proof-of-stake blockchains with dynamic availability. In D. Lie, M. Mannan, M. Backes, and X. Wang, editors, *ACM CCS 2018*, pages 913–930. ACM Press, Oct. 2018.

[13] Such an additional layer is declared as future work in [9].

3. C. Badertscher, U. Maurer, D. Tschudi, and V. Zikas. Bitcoin as a transaction ledger: A composable treatment. In J. Katz and H. Shacham, editors, *CRYPTO 2017, Part I*, volume 10401 of *LNCS*, pages 324–356. Springer, Heidelberg, Aug. 2017.
4. I. Bentov, J. Loss, T. Moran, and B. Shani. The spacemesh protocol: Tortoise and hare consensus...in...space. White paper, Version 1.0.2, 2019. `https://spacemesh.io/`.
5. E. Blum, A. Kiayias, C. Moore, S. Quader, and A. Russell. The combinatorics of the longest-chain rule: Linear consistency for proof-of-stake blockchains. In S. Chawla, editor, *31st SODA*, pages 1135–1154. ACM-SIAM, Jan. 2020.
6. J. Bonneau. Hostile blockchain takeovers (short paper). In A. Zohar, I. Eyal, V. Teague, J. Clark, A. Bracciali, F. Pintore, and M. Sala, editors, *Financial Cryptography and Data Security - FC 2018 International Workshops, BITCOIN, VOTING, and WTSC, Nieuwpoort, Curaçao, March 2, 2018, Revised Selected Papers*, volume 10958 of *Lecture Notes in Computer Science*, pages 92–100. Springer, 2018.
7. V. Buterin. A next-generation smart contract and decentralized application platform, 2013. `https://github.com/ethereum/wiki/wiki/White-Paper`.
8. M. Castro, B. Liskov, et al. Practical byzantine fault tolerance. In *OSDI*, volume 99, pages 173–186, 1999.
9. J. Chen and S. Micali. Algorand: A secure and efficient distributed ledger. *Theor. Comput. Sci.*, 777:155–183, 2019.
10. P. Daian, R. Pass, and E. Shi. Snow white: Robustly reconfigurable consensus and applications to provably secure proof of stake. In *Financial Cryptography and Data Security - 23rd International Conference, FC 2019, Frigate Bay, St. Kitts and Nevis, February 18-22, 2019, Revised Selected Papers*, pages 23–41, 2019.
11. B. David, P. Gazi, A. Kiayias, and A. Russell. Ouroboros praos: An adaptively-secure, semi-synchronous proof-of-stake blockchain. In J. B. Nielsen and V. Rijmen, editors, *EUROCRYPT 2018, Part II*, volume 10821 of *LNCS*, pages 66–98. Springer, Heidelberg, Apr. / May 2018.
12. Y. Dodis and A. Yampolskiy. A verifiable random function with short proofs and keys. In S. Vaudenay, editor, *PKC 2005*, volume 3386 of *LNCS*, pages 416–431. Springer, Heidelberg, Jan. 2005.
13. C. Dwork, N. A. Lynch, and L. J. Stockmeyer. Consensus in the presence of partial synchrony (preliminary version). In R. L. Probert, N. A. Lynch, and N. Santoro, editors, *3rd ACM PODC*, pages 103–118. ACM, Aug. 1984.
14. I. Eyal and E. G. Sirer. Majority is not enough: Bitcoin mining is vulnerable. In *International conference on financial cryptography and data security*, pages 436–454. Springer, 2014.
15. P. Feldman and S. Micali. Byzantine agreement in constant expected time (and trusting no one). In *26th FOCS*, pages 267–276. IEEE Computer Society Press, Oct. 1985.
16. M. Fitzi. *Generalized communication and security models in Byzantine agreement.* PhD thesis, ETH Zurich, Zürich, Switzerland, 2003.
17. P. Gaži, A. Kiayias, and A. Russell. Tight consistency bounds for bitcoin. Cryptology ePrint Archive, Report 2020/661, 2020. `https://eprint.iacr.org/2020/661`.
18. J. A. Garay and A. Kiayias. Sok: A consensus taxonomy in the blockchain era. In S. Jarecki, editor, *Topics in Cryptology - CT-RSA 2020 - The Cryptographers' Track at the RSA Conference 2020, San Francisco, CA, USA, February 24-28, 2020, Proceedings*, volume 12006 of *Lecture Notes in Computer Science*, pages 284–318. Springer, 2020.
19. J. A. Garay, A. Kiayias, and N. Leonardos. The bitcoin backbone protocol: Analysis and applications. In E. Oswald and M. Fischlin, editors, *EUROCRYPT 2015, Part II*, volume 9057 of *LNCS*, pages 281–310. Springer, Heidelberg, Apr. 2015.
20. J. A. Garay, A. Kiayias, and N. Leonardos. The bitcoin backbone protocol with chains of variable difficulty. In J. Katz and H. Shacham, editors, *CRYPTO 2017, Part I*, volume 10401 of *LNCS*, pages 291–323. Springer, Heidelberg, Aug. 2017.
21. C. E. Kelso. Bitcoin gold hacked for $18 million, 2018. `https://news.bitcoin.com/bitcoin-gold-hacked-for-18-million/`.
22. A. Kiayias, E. Koutsoupias, M. Kyropoulou, and Y. Tselekounis. Blockchain mining games. In V. Conitzer, D. Bergemann, and Y. Chen, editors, *Proceedings of the 2016 ACM Conference on Economics and Computation, EC '16, Maastricht, The Netherlands, July 24-28, 2016*, pages 365–382. ACM, 2016.
23. A. Kiayias, A. Russell, B. David, and R. Oliynykov. Ouroboros: A provably secure proof-of-stake blockchain protocol. In J. Katz and H. Shacham, editors, *CRYPTO 2017, Part I*, volume 10401 of *LNCS*, pages 357–388. Springer, Heidelberg, Aug. 2017.
24. L. Kiffer, R. Rajaraman, and a. shelat. A better method to analyze blockchain consistency. In D. Lie, M. Mannan, M. Backes, and X. Wang, editors, *ACM CCS 2018*, pages 729–744. ACM Press, Oct. 2018.

25. A. Klenke and L. Mattner. Stochastic ordering of classical discrete distributions. *Advances in Applied Probability*, 42(2):392–410, 2010.
26. J. Knockel, G. Saad, and J. Saia. Self-healing of byzantine faults. In T. Higashino, Y. Katayama, T. Masuzawa, M. Potop-Butucaru, and M. Yamashita, editors, *Stabilization, Safety, and Security of Distributed Systems - 15th International Symposium, SSS 2013, Osaka, Japan, November 13-16, 2013. Proceedings*, volume 8255 of *Lecture Notes in Computer Science*, pages 98–112. Springer, 2013.
27. J. Kwon. Tendermint : Consensus without mining. 2014.
28. M. Mitzenmacher and E. Upfal. *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge University Press, 2005.
29. S. Nakamoto. Bitcoin: A peer-to-peer electronic cash system, 2008. `http://bitcoin.org/bitcoin.pdf`.
30. M. Nesbitt. Deep chain reorganization detected on ethereum classic (etc), 2019. `https://blog.coinbase.com/ethereum-classic-etc-is-currently-being-51-attacked-33be13ce32de`.
31. S. O'Neal. Bitcoin cash hard fork battle: Who is winning the hash war, 2018. `https://cointelegraph.com/news/bitcoin-cash-hard-fork-battle-who-is-winning-the-hash-war`.
32. G. Pandurangan and A. Trehan. Xheal: localized self-healing using expanders. In C. Gavoille and P. Fraigniaud, editors, *Proceedings of the 30th Annual ACM Symposium on Principles of Distributed Computing, PODC 2011, San Jose, CA, USA, June 6-8, 2011*, pages 301–310. ACM, 2011.
33. R. Pass, L. Seeman, and a. shelat. Analysis of the blockchain protocol in asynchronous networks. In J.-S. Coron and J. B. Nielsen, editors, *EUROCRYPT 2017, Part II*, volume 10211 of *LNCS*, pages 643–673. Springer, Heidelberg, Apr. / May 2017.
34. R. Pass and E. Shi. The sleepy model of consensus. In T. Takagi and T. Peyrin, editors, *ASIACRYPT 2017, Part II*, volume 10625 of *LNCS*, pages 380–409. Springer, Heidelberg, Dec. 2017.
35. D. Pollard. Miniempirical, 2015. `http://www.stat.yale.edu/~pollard/Books/Mini/`.
36. J. Redman. Vertcoin network suffers 300-block reorg following 51% attacks, 2018. `https://news.bitcoin.com/vertcoin-network-51-attacked-and-suffers-from-a-reorg-300-blocks-deep/`.
37. L. Ren. Analysis of Nakamoto consensus. Cryptology ePrint Archive, Report 2019/943, 2019. `https://eprint.iacr.org/2019/943`.
38. J. Saia and A. Trehan. Picking up the pieces: Self-healing in reconfigurable networks. In *22nd IEEE International Symposium on Parallel and Distributed Processing, IPDPS 2008, Miami, Florida USA, April 14-18, 2008*, pages 1–12. IEEE, 2008.
39. A. Sapirshtein, Y. Sompolinsky, and A. Zohar. Optimal selfish mining strategies in bitcoin. In *International Conference on Financial Cryptography and Data Security*, pages 515–532. Springer, 2016.
40. M. Yin, D. Malkhi, M. K. Reiter, G. G. Gueta, and I. Abraham. Hotstuff: Bft consensus with linearity and responsiveness. In *Proceedings of the 2019 ACM Symposium on Principles of Distributed Computing*, PODC '19, pages 347–356, New York, NY, USA, 2019. Association for Computing Machinery.
41. ZenCash. Zencash statement on double spend attack, 2018. `https://blog.zencash.com/zencash-statement-on-double-spend-attack/`.

A On the Chain-Growth Argument for Nakamoto-Style Blockchains

We review here the chain-growth arguments for the proof-of-work protocols (e.g., Bitcoin) which are well-studied in the literature and put it into our context (as invoked in Theorem 1): (S, α)*-chain growth* is a property of an execution of a blockchain protocol asserting that any chain held by an observer of the protocol as a result of applying the longest-chain rule contains at least αS blocks generated during any past time interval of length S. Unlike consistency, chain growth can be inferred in a straightforward manner from classical tail bounds for i.i.d. random variables. In our setting (S, α)-chain growth is guaranteed with probability $1 - \exp(-\Omega(S))L^2$, so long as α is a small enough constant (which depends on an assumed lower bound such as n_0 in our model); here L refers to the lifetime of the protocol and the constant hidden by the asymptotic notation is determined by Δ and the hashing power of the adversarial and honest participants.

To survey the proof, with any chain C violating this condition—by exhibiting too few blocks in an interval $\mathcal{I}$ of length S—we may associate a "witness region" $\mathcal{R}$ determined by the last honest block prior to $\mathcal{I}$ and first honest block after $\mathcal{I}$ (with the understanding that the region is terminated by the last block of C if there is no such following honest block). The total number of blocks in the interior of this witness region (that is, excluding the first and last blocks) is certainly no more than $\alpha S + A_{\mathcal{R}}$, where the first term accounts for the blocks in $\mathcal{I}$ and the second accounts for the total number of (adversarial) blocks in $\mathcal{R}$. On the other hand, beginning with the block B and considering the pattern of slots in $\mathcal{R}$ with honest hashing successes, one can obtain an immediate (lower) bound on the length of the chain broadcast by the last honest player with a hashing success Δ slots prior to the end of $\mathcal{R}$ (and hence in time to be seen by the observer). The length of this chain cannot exceed that of C, by assumption.

To complete the argument, one shows that for any fixed witness region $\mathcal{R}$ it is very unlikely for $\alpha S + A_{\mathcal{R}}$ to exceed the additional length that the honest players alone in $\mathcal{R}$ must have added to any honestly broadcast chain prior to $\mathcal{R}$; by the discussion above, this rules out the possibility of a chain growth violation spanning $\mathcal{R}$. The quantity $A_{\mathcal{R}}$ is stochastically dominated by a simple sum of i.i.d. random variables (corresponding to the adversary's maximal hashing power) and thus can be bounded by a standard Chernoff bound. As for the length of the longest honestly broadcast chain available to the observer, one notes that any slot containing an honest hashing success must necessarily increase by at least 1 the length of the longest chain thus far broadcast by honest participants when compared against those chains broadcast Δ slots prior. This lower bound can likewise be analyzed by standard tail bounds (e.g., the McDiarmid or Azuma inequality), yielding the error terms discussed above or alternatively again by a sum of i.i.d. random variables of a worst-case reference experiment. Note that the L^2 term arises from a union bound taken over all initial and final positions of the interval $\mathcal{R}$. For details of these approaches, using the McDiarmid inequality or a lower bound based on simple reference experiment, we refer to [17,3,33].