Ledger Combiners for Fast Settlement

Matthias Fitzi[1], Peter Gaži[1], Aggelos Kiayias[1,2], and Alexander Russell[1,3]

[1] IOHK Research
[2] University of Edinburgh
[3] University of Connecticut
firstname.lastname@iohk.io

Abstract. Blockchain protocols based on variations of the longest-chain rule—whether following the proof-of-work paradigm or one of its alternatives—suffer from a fundamental latency barrier. This arises from the need to collect a sufficient number of blocks on top of a transaction-bearing block to guarantee the transaction's stability while limiting the rate at which blocks can be created in order to prevent security-threatening forks. Our main result is a black-box security-amplifying combiner based on parallel composition of m blockchains that achieves $\Theta(m)$-fold security amplification for conflict-free transactions or, equivalently, $\Theta(m)$-fold reduction in latency. Our construction breaks the latency barrier to achieve, for the first time, a ledger based purely on Nakamoto longest-chain consensus guaranteeing worst-case constant-time settlement for conflict-free transactions: settlement can be accelerated to a constant multiple of block propagation time with negligible error.
Operationally, our construction shows how to view any family of blockchains as a unified, virtual ledger without requiring any coordination among the chains or any new protocol metadata. Users of the system have the option to inject a transaction into a single constituent blockchain or—if they desire accelerated settlement—all of the constituent blockchains. Our presentation and proofs introduce a new formalism for reasoning about blockchains, the *dynamic ledger*, and articulate our constructions as transformations of dynamic ledgers that amplify security. We also illustrate the versatility of this formalism by presenting robust-combiner constructions for blockchains that can protect against complete adversarial control of a minority of a family of blockchains.

1 Introduction

Since the appearance of Bitcoin [33] in 2009, dozens of projects from both academia and industry have proposed protocols for maintaining decentralized, robust transaction ledgers in a permissionless setting. The prominent design paradigm in this space comes from the Bitcoin protocol itself, often referred to as "Nakamoto-style" ledger consensus. This approach adopts the *blockchain*—a linearly ordered sequence of blocks, each of which commits to the previous history and may contain new transactions—as the fundamental data structure for maintaining the ledger. The core consensus algorithm then calls for eligible

protocol participants to create transaction-bearing blocks, append them to the longest chain they observe, and broadcast the result; this implicitly declares a "vote" for a unique ordered sequence of past transactions—the ledger. As a result, the immutability of a particular portion of the ledger is not immediate, but rather grows gradually with the number of blocks (representing votes) amassed on top of it in the blockchain. This paradigm has been featured in both theoretical proposals as well as deployed systems and can be instantiated with a wide variety of Sybil-resistant mechanisms such as proof of work (Bitcoin, Ethereum [7] and a vast majority of deployed blockchains), proof of stake [24,11,12,2,3], proof of space [34,10], and others.

In terms of performance, one of the key measures of interest for any distributed ledger protocol is *latency*, also called *settlement time*. Roughly speaking, this is the time elapsed between the moment a signed transaction is injected into the protocol and the time it becomes universally recognized as immutable. While Nakamoto-style consensus protocols have attracted attention both for their simplicity and for various desirable security features,[4] they appear to face a fundamental barrier when it comes to latency. Informally, for a transaction to become accepted as stable, a sufficient number of blocks (representing an agreement over a representative fraction of the parties, weighted according to the Sybil-resistant mechanism in place) must be collected on top of the block containing this transaction. However, blocks can only be created at a limited rate dictated by the delays introduced by the underlying communication network: if blocks are routinely created by participants that have not yet received recent previous blocks, forks in the blockchain appear even without adversarial interference. These forks then result in a division of the honest majority and represent a threat to the protocol's security. This relationship is now quite well understood [35].

One way to address this disadvantage without giving up on the Nakamoto paradigm (and its advantages) is to carefully design overlay structures on top of the plain Nakamoto-style blockchain. Several such proposals exist, and can be roughly split into two categories. The first group of proposals (e.g., [36,38,27]) still produces a full ledger of all settled transactions, but relies on stronger assumptions for their latency improvement, such as a higher threshold of honest participants. The second category are so-called layer-2 designs implementing payment [41,14] or state [15] channels that only need limited interaction with the slow blockchain; however, they divert from the original goal of maintaining a distributed ledger of all executed transactions. Hence, the following fundamental question remains:

What is the fastest achievable settlement time for Nakamoto-style consensus?

This question has also been recently addressed by elegant concurrent work on the Prism protocol [5], albeit with somewhat different goals; we give a detailed comparison between our work and [5] in Section 1.2.

[4] For example, they can provide security in the Byzantine setting with simple honest majority [9,36,38], and resilience against fluctuating participation [37,2].

1.1 Our Contributions

We approach the challenge of designing low-latency ledgers by introducing a black-box technique for "combining" a family of existing ledgers into a new, virtual ledger that provides amplified security properties. Our technique results in a system with striking simplicity: The construction gives *a deterministic rule for interpreting an arbitrary family of m constituent ledgers as a single virtual ledger*. Participants of the system maintain their current view of each constituent ledger and, via this interpretation, a view of the master combined ledger. Users simply inject their transactions into the constituent ledgers as usual. We show that when users inject transactions into a single constituent ledger they are provided with settlement guarantees (in the virtual ledger) roughly consistent with those offered by the constituent ledgers. On the other hand, when a conflict-free transaction is injected into all of the constituent ledgers, it enjoys a $1/\Theta(m)$ *multiplicative improvement in settlement time*. Of course, settlement time cannot be reduced beyond the time required for a block to be transmitted across the network; however, our results adapt smoothly to this limit; in particular, by taking m to scale with the security parameter of the system, we obtain $O(1)$ settlement time for conflict-free transactions (except with negligible probability). We remark that in cryptocurrency ledgers, such as Bitcoin, transaction issuers always have the option to submit conflict-free transactions so that the assumption is not a limitation. While the results do not require any specific coordination among the ledgers, they naturally require a measure of stochastic independence; we discuss this in detail below.

We present our results by formulating an abstract notion which we call a *dynamic ledger*. Our constructions transform a family of such dynamic ledgers into an associated dynamic ledger (as indicated above) in a way that amplifies the security properties. Typical blockchain algorithms are direct instantiations of this abstraction: our techniques can thus be applied in wide generality to existing blockchains such Bitcoin, Ethereum, Ouroboros, etc.

Such a transformation is a "combiner" in the classical cryptographic sense of the word: an operator for cryptographic primitives that acts in a black-box manner on a number of underlying implementations of a primitive with the objective of realizing a strengthened implementation of the same primitive. This folklore idea in cryptography first received an explicit treatment by Herzberg [23]. One of the objectives for developing combiners—especially prominent in the context of hash functions—was the concept of *robustness*. In particular, a robust combiner maintains the security of the combined implementation despite the security failure of any number (up to a threshold) of the underlying input implementations. Another objective for developing combiners is *amplification*: In an amplification combiner, the goal is to improve a certain security property of the combined implementation to a level that goes significantly beyond the security offered by the underlying input implementations. The combiner discussed above is of the amplification variety; later in the paper, we also show how to achieve robustness in our setting.

With this summary behind us, we describe our contributions in more detail.

A Model for Abstract Ledgers. We provide a new mathematical abstraction of a distributed ledger that can be used to reflect an arbitrary ledger protocol, but is particularly well-suited for describing Nakamoto-style blockchains with eventual-consensus behavior (regardless of their underlying Sybil-resistant election mechanism). Its main design goals are generality and simplicity, so as to allow for a clean study of generic constructions with such ledgers that is unencumbered by the execution details of the underlying protocols.

Roughly speaking, our abstraction—called a *dynamic ledger*—determines at every point in time (i) a set of transactions that are contained in the ledger; and (ii) a mapping that assigns to each transaction a real value called its *rank*. The rank plays several roles: it is used to order the transactions in the ledger, describe their stability, and maintain a loose connection to actual time; the most natural example of a rank is the timestamp of the transaction's block in Bitcoin. (In fact, a simple monotonicity transform is necessary; see Section 4.2.)

A dynamic ledger satisfies three fundamental properties: *liveness*, *absolute persistence*, and *relative persistence*. The former two properties are direct analogues of the well-established notions of persistence and liveness introduced by previous formalizations of blockchain protocols; the notion of relative persistence is novel. In a nutshell, it is a weakening of absolute persistence that guarantees that the rank of a transaction cannot significantly change in the future; in particular the relative order of the transaction with respect to sufficiently distant transactions is determined. This is particularly useful for reasoning about transaction settlement in the typical setting of interest: when transaction validity depends only on its ordering with respect to *conflicting* transactions. Looking ahead, relative persistence is exactly the notion that allows us to achieve the full benefits of our amplification combiner; it appears to be of independent interest as well, as it also arises naturally in our robust combiner.

A combiner for consistency amplification and latency reduction; the combined rank function. Our main technical contribution, discussed briefly above, is an amplification combiner for latency reduction of abstract ledgers. This combiner builds a "combined ledger" (or virtual ledger) as a deterministic function of m underlying dynamic ledgers. Participants insert their transaction into any number of the underlying ledgers, depending on the desired settlement-time guarantees.

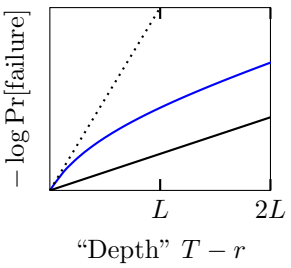
The major challenge is the definition and analysis of the combiner rank function. Rank is an abstract notion of position in the ledger that is tethered to absolute time by the security guarantees: for example, in a ledger at time T the probability that a transaction appearing at rank r is later disrupted is a function of $T - r$; the standard case, where the underlying ledgers provide "linear consistency," guarantees consistency error $\exp(-\Omega(T - r))$. Note that, in general, there is no guarantee that transactions will appear in all underlying ledgers so the combined rank function must somehow assign rank in a fashion that appropriately reflects both deep transactions appearing in a single ledger and shallower transactions appearing in many ledgers. This state of affairs introduces two conflicting goals: in order to achieve linear amplification we insist that when a

transaction appears in all m ledgers, our constructed ledger yields settlement error $\exp(-\Omega(m(T-r)))$—note the factor of m in the exponent; on the other hand, a transaction appearing in a single ledger will be assigned some finite rank and thus for large values of T we cannot hope to beat $\exp(-\Omega(T-r))$, the consistency guarantee of a single ledger. To realize this, our construction (and combined rank function) is determined by a parameter L which, intuitively, determines the transition between these two regimes. One should think of L proportional to the security parameter of the system, so that $2^{-\Theta(L)}$ is an acceptable bound for undesirable events; thus, injecting a transaction into all the ledgers achieves this $2^{-\Theta(L)}$ security bound $\Theta(m)$ times faster than transactions submitted to a single ledger.

It is a rather remarkable fact that the behavior we demand is provided by the exponential weighting functions that arise naturally in the theory of regret minimization (e.g., the multiplicative weights algorithm [1]). The actual form of our combined rank function is

$$\exp(-\mathsf{combinedRank}(\mathrm{tx})/L) = \frac{1}{m}\sum_{i=1}^{m}\exp(-\mathsf{rank}_i(\mathrm{tx})/L)\,.$$

The (log scale) consistency error achieved by this rank function, when coupled with underlying ledgers that offer linear consistency, is informally illustrated by the blue line in the figure below. The solid black line is the consistency error offered by the underlying ledgers; one can clearly see the region of rapid growth (prior to L) followed by the region where the slope stabilizes to that of the single ledger bounds, as it must. The dotted line has slope exactly m times that of the "single ledger" line, corresponding intuitively to "perfect amplification."



We analyze two extreme scenarios and show that while insertion of a transaction into a single ledger leads to a settlement time comparable to the one provided by the underlying ledgers, inserting the transaction into all m ledgers results in a speed-up by a linear factor $\Theta(m)$. In the natural setting where there is a cost associated with including a transaction in each ledger, we emphasize that the construction yields a trade-off between transaction fee and settlement time: transactions appearing in more chains settle faster. The choice can be made on a per-transaction basis by its sender. Moreover, by considering a sufficient number of parallel chains m, this allows us to achieve relative settlement in constant time except with negligible error.

Clearly, amplification-type results can only be obtained under some sort of independence assumption on the underlying ledgers. We characterize a generic (black-box) assumption, called *subindependence*, which is weaker than full independence of the ledgers and sufficient for our results. We also show how subinde-

pendence can be naturally achieved by existing techniques in both proof-of-work and proof-of-stake settings; details appear in Section 4.1.

Our construction does not require any coordination between the underlying ledgers, it can be deployed on top of existing blockchains without direct cooperation from parties maintaining the ledgers, so long as these ledgers maintain their persistence and liveness guarantees, are sufficiently independent, and allow for inclusion of a sufficiently general class of transaction data.

Finally, we show how our construction can be applied to the most familiar setting of proof-of-work (PoW) blockchains. Specifically, applying our combiner to $m = \lambda$ PoW blockchains yields a construction C providing constant-time relative settlement except with probability negligible in λ, articulated in Theorem 1 below. For concreteness, we work in the synchronous (p, q)-*flat PoW model* that assumes the existence of n parties, each of which is allowed to issue q PoW queries per round that independently succeed with probability p (see, e.g., [20] for details).

Theorem 1 (Informal). *Let $\epsilon > 0$ and let λ denote the security parameter. There exists a construction C that, if executed in the synchronous (p, q)-flat PoW model with n parties out of which at least a $(1/2 + \varepsilon)$-fraction is honest, achieves relative settlement in time $O(1)$ except with an error probability negligible in λ.*

Hidden in the asymptotic description above is the dependence of p, q, and n on the security parameter λ which must in fact satisfy some natural conditions. We give a formal statement corresponding to Theorem 1, together with a precise description of the construction C, as Corollary 7 in Section 4.3.

A simplified illustration of the settlement speed-up provided by our construction is given in the Appendix of the full paper [17].

A Robust Ledger Combiner. As our final contribution, we describe a class of constructions of robust ledger combiners: a black-box construction on top of m ledgers that maintains relative persistence and liveness guarantees even if the contents of a δ-fraction of these ledgers (chosen adaptively) are arbitrarily corrupted, for δ up to $1/2$. The individual constructions in this class are parametrized by the choice of an estimator function that is a part of the combiner's rank function; we show that the concrete choice of this estimator represents a trade-off between δ and the *stability* of the combiner, a metric of how much the ranks of individual transactions in the combiner change as a result of a corruption respecting the δ-threshold. This construction serves as an additional illustration of the generality of our ledger abstraction. We refer the reader to the full paper [17] for the details.

1.2 Related Work

The formal modeling of robust transaction ledgers and blockchain protocols goes back to the property-based analysis of Bitcoin due to Garay et al. [20] and Pass et al. [35]. These works identified the central properties of common prefix, chain growth, and chain quality and demonstrated how they imply the desired persistence and liveness of the resulting ledger. A composable analysis of a blockchain protocol (namely Bitcoin) in the UC framework [8] along with the

realized ledger functionality first appeared in [4] and, later, essentially the same functionality was shown to be realized by proof-of-stake protocols in [2,3].

The notion of combiners was formally proposed in [23]. Robust combiners for hash functions were further studied in [6,40] and also applied to other primitives such as oblivious transfer [22]. Amplification combiners were introduced by [16] who also observed that classical results in security amplification can be seen as such combiners. Indistinguishability amplification for random functions and permutations achieved by certain combiners from a class of so-called *neutralizing* constructions was studied both in the information-theoretic [43,29,28,30,21] and computational [26,32,39,13,31] settings.

Various approaches are known to reduce settlement times of Nakamoto-style blockchains. One approach is to deviate from the single-chain structure, arranging blocks in a directed acyclic graph (DAG) as first suggested by Lerner [25]. Sompolinsky et al. [42] gave a DAG-based construction that substantially reduces settlement times at the expense of giving up on a total order on all transactions in the ledger. Another approach explores "hybrid" protocols where committee-based consensus reduces latency in the optimistic case [36,38]. In context of proof-of-stake, Algorand [9] reduces settlement times over eventual-consensus proof-of-stake protocols by finalizing each block via a Byzantine Agreement subprotocol before moving to the next one. However, Algorand cannot tolerate fluctuating participation or adversarial stake ratio up to 1/2. Moreover, its constant-time settlement guarantees are only provided *in expectation*, in contrast to our worst-case guarantees.

Concurrent work on the Prism protocol [5] also addressed the efficiency of Nakamoto consensus. (We remark that a preliminary version of this paper [18] was published as an IACR eprint in 2018.) Prism is a concrete, PoW-based ledger protocol optimizing both throughput and latency compared to Bitcoin. Prism similarly approaches the problem by introducing "parallel blockchains," though in a different form. Our approach has some notable advantages in comparison with Prism: (i) our construction is generic and can be deployed on top of existing ledgers with arbitrary Sybil-resistant mechanisms; (ii) we provide worst-case constant-time settlement except with a negligible error probability while Prism (similarly to Algorand) only provides *expected* constant-time settlement; (iii) we base our results on the generic subindependence assumption that is weaker than full independence, which is assumed in Prism (though not achieved by their PoW mechanism). On the other hand, Prism has an important feature which clearly sets it apart from our work: it explicitly models and optimizes throughput. A more detailed comparison between our work and Prism is given in our full paper.

2 The Ledger Abstraction

In this section we define *abstract ledgers* which describe the functionality provided by distributed ledger protocols such as Bitcoin. Our goal here is to capture this behavior in an abstract, high-level manner, which allows us to express our composition results unencumbered by the details of the individual protocols.

2.1 Ledgers and Dynamic Ledgers

We start by defining an abstraction of an individual snapshot of the state of a ledger protocol, which we call a *ledger*. A ledger reflects a collection of *transactions* which are given a linear order by way of a general function called *rank*. As a basis for intuition about our definitions and proofs, we mention that, roughly speaking, Bitcoin realizes such a ledger where the rank function is given by the timestamp corresponding to the block containing the transaction; we give a more detailed discussion in Section 4.2.

Our ledger will operate over a *transaction space* which we define first.

Definition 1 (Transaction space). *A* transaction space *is a pair* $(\mathcal{T}, \prec_{\mathcal{T}})$*, where* $\mathcal{T}$ *is a set of "transactions" and* $\prec_{\mathcal{T}}$ *is a linear order on* $\mathcal{T}$*. A* conflict relation C *on a transaction space* $\mathcal{T}$ *is a symmetric binary relation on* $\mathcal{T}$*; if* $(\mathrm{tx}_1, \mathrm{tx}_2) \in C$ *for two transactions* $\mathrm{tx}_1, \mathrm{tx}_2 \in \mathcal{T}$*, we say that* tx_1 conflicts with tx_2*, we write* $\mathrm{conflict}(\mathrm{tx}) \subseteq \mathcal{T}$ *for the set of transactions conflicting with* tx*.*

The linear order $\prec_{\mathcal{T}}$ on the ambient transaction space $\mathcal{T}$ is largely incidental; it is only used in our setting to break ties among transactions with a common rank. Thus, in practice this linear order can be instantiated with a simple "syntactic" property—such as a lexicographic ordering—rather than an ordering that reflects any semantics about the transactions.

In contrast, if a transaction space is equipped with a conflict relation, this is intended to carry semantic value; in a conventional UTXO transaction model (such as that of many deployed blockchains) two transactions conflict if they share UTXO inputs. As we discuss below, a conflict structure permits a more flexible notion of settlement that is only required to provide strong guarantees for non-conflicting transactions.

Definition 2 (Ledger). *A* ledger $\mathbf{L}$ *for a transaction space* $(\mathcal{T}, \prec_{\mathcal{T}})$ *is a pair* (T, rank) *where:* $T \subseteq \mathcal{T}$ *is a subset of* transactions *and* $\mathsf{rank}\colon \mathcal{T} \to \mathbb{R}^+ \cup \{\infty\}$ *is a function taking finite values precisely on the set* T*; that is,* $T = \{\mathrm{tx} \in \mathcal{T} \mid \mathsf{rank}(\mathrm{tx}) \neq \infty\}$*. The value* $\mathsf{rank}(\mathrm{tx})$ *is referred to as the* rank *of the transaction* tx*. Notationally, if* $\mathbf{L}$ *is a ledger we routinely overload the symbol* $\mathbf{L}$ *to stand for its set of transactions (the above set* T*).*

The linear order $\prec_{\mathcal{T}}$ *and the rank function* $\mathsf{rank}(\cdot)$ *induce a linear order* $\prec_{\mathbf{L}}$ *on the ledger by the rule*

$$x \prec_{\mathbf{L}} y :\Leftrightarrow (\mathsf{rank}(x) < \mathsf{rank}(y) \vee (\mathsf{rank}(x) = \mathsf{rank}(y) \wedge x \prec_{\mathcal{T}} y)) \ .$$

(Thus the underlying total order $\prec_{\mathcal{T}}$ *is only used to "break ties.")*

For a ledger $\mathbf{L} = (T, \mathsf{rank})$ *and a threshold* r*, we let* $\mathbf{L}\lceil r \rceil \stackrel{\text{def}}{=} (T', \mathsf{rank}')$ *denote the ledger consisting of transactions* $T' \stackrel{\text{def}}{=} \{\mathrm{tx} \in T \mid \mathsf{rank}(\mathrm{tx}) \leq r\}$ *with the inherited rank function:* $\mathsf{rank}'(\mathrm{tx}) = \mathsf{rank}(\mathrm{tx})$ *for all* $\mathrm{tx} \in T'$ *and equal to* ∞ *otherwise. Similarly, for a transaction* $\mathrm{tx} \in \mathbf{L}$*, let* $\mathbf{L}\lceil \mathrm{tx} \rceil$ *denote the ledger* $\{\mathrm{tx}' \mid \mathrm{tx}' \preceq_{\mathbf{L}} \mathrm{tx}\}$ *with the inherited rank function.*

The above notion of a ledger captures a static state; we extend it to describe evolution in time as follows.

Definition 3 (Dynamic ledger). *Consider the sequence of time slots $t \in \mathbb{N}$ and any sequence of sets of transactions $A^{(0)}, A^{(1)}, \ldots$ (each a subset of a common transaction space $\mathcal{T}$) denoting the transactions that* arrive *at each time slot. A* dynamic ledger *is a sequence of random variables $\mathbf{D} \stackrel{\text{def}}{=} \mathbf{L}^{(0)}, \mathbf{L}^{(1)}, \ldots$, that satisfy the following properties parametrized by security functions $\mathrm{p}_R^+ : (\mathbb{R}^+)^2 \to [0,1]$ and $\mathrm{p}_R^-, \mathrm{p}_A, \mathrm{l} : \mathbb{R}^+ \to [0,1]$:*

Liveness. *For every $r \geq 0$, $t_0 \geq 0$, and $t \geq t_0 + r$,*

$$\Pr\left[L_{r,t_0,t}\right] \stackrel{\text{def}}{=} \Pr\left[A^{(t_0)} \not\subseteq \mathbf{L}^{(t)}\left\lceil t_0 + r\right\rceil\right] \leq \mathrm{l}(r)\,.$$

Absolute Persistence. *For each rank $r \geq 0$, time $t_0 \geq 0$, and $t \geq t_0$, we have $\mathbf{L}^{(t_0)}\left\lceil t_0 - r\right\rceil = \mathbf{L}^{(t)}\left\lceil t_0 - r\right\rceil$ except with small failure probability. Specifically, for all $r, t_0 \geq 0$,*

$$\Pr\left[P_{r,t_0}\right] \stackrel{\text{def}}{=} \Pr\left[\exists t \geq t_0, \mathbf{L}^{(t_0)}\left\lceil t_0 - r\right\rceil \neq \mathbf{L}^{(t)}\left\lceil t_0 - r\right\rceil\right] \leq \mathrm{p}_A(r)\,.$$

Relative Persistence. *For each $r^+, r^- \geq 0$, time $t_0 \geq 0$, and $t \geq t_0$, we have $\mathbf{L}^{(t_0)}\left\lceil t_0 - r^- - r^+\right\rceil \subseteq \mathbf{L}^{(t)}\left\lceil t_0 - r^-\right\rceil \subseteq \mathbf{L}^{(t_0)}$ except with small failure probability. Specifically, for each r^-, r^+, $t_0 \geq 0$:*

$$\Pr\left[\exists t \geq t_0, \mathbf{L}^{(t_0)}\left\lceil t_0 - r^- - r^+\right\rceil \not\subseteq \mathbf{L}^{(t)}\left\lceil t_0 - r^-\right\rceil\right] \leq \mathrm{p}_R^+(r^-, r^+)\,,$$
$$\Pr\left[\exists t \geq t_0, \mathbf{L}^{(t)}\left\lceil t_0 - r^-\right\rceil \not\subseteq \mathbf{L}^{(t_0)}\right] \leq \mathrm{p}_R^-(r^-)\,.$$

As indicated, we let P_{r,t_0} and $L_{r,t_0,t}$ denote the absolute-persistence failure event with parameters (r, t_0) and the liveness failure event with parameters (r, t_0, t), respectively.

The above definition deserves a detailed discussion. A dynamic ledger is a sequence of ledgers—one for each time slot t—which reflects the current state of the ledger structure $\mathbf{L}^{(t)}$ at that time. Throughout the paper, we will use the superscript notation $\cdot^{(t)}$ to denote the time coordinate.

Absolute persistence and *liveness* capture the standard design features of distributed ledger protocols: absolute persistence mandates that at time t_0, the state of the ledger up to rank $t_0 - r$ is fixed for all future times, except with error $\mathrm{p}_A(r)$. Liveness, on the other hand, guarantees that any transaction appearing in $A^{(t_0)}$ will be a part of a (later) ledger at time $t \geq t_0 + r$ with a rank at most $t_0 + r$, except with error at most $\mathrm{l}(r)$. Note that the liveness guarantee only pertains to transactions tx appearing in the sets $A^{(t_0)}$, which may not necessarily "explain" all of the transactions in the ledger; in particular, we do not always insist that $\mathbf{L}^{(t)} \subseteq \bigcup_{s \leq t} A^{(s)}$. This extra flexibility permits us to simultaneously study differing liveness guarantees for various subclasses of transactions processed by a particular ledger (see Sect. 3.2).

The remaining property, *relative persistence*, is more complex: It is a weakening of absolute persistence by *not* requiring future stability for the prefix of the currently seen ledger $\mathbf{L}^{(t_0)}$ up to rank $t_0 - r^- - r^+$; it merely asks that no transaction tx *currently contained* in it will rise to a rank exceeding $t_0 - r^-$; likewise, it insists that no transaction tx' currently absent in the ledger will ever achieve a rank below $t_0 - r^-$, potentially overtaking tx. This property bears a direct connection to the notion of transaction settlement as we discuss in Section 2.3. Looking ahead, we note that relative persistence provides sufficient guarantees for settling transactions that are only invalidated by "conflicting" transactions, and our combiner will achieve stronger relative-persistence than absolute-persistence guarantees, allowing for our latency-reduction results in Section 3.

Note that absolute persistence for some r clearly implies relative persistence with $r^+ = 0$ and $r^- = r$. A natural parametrization that makes our notions meaningful is where each $f \in \left\{\mathrm{p}_A, \mathrm{p}_R^-, 1\right\}$ is monotonically decreasing and satisfies $f(0) \geq 1$ (similarly, p_R^+ should be monotonically decreasing in each coordinate and $\mathrm{p}_R^+(r^-, r^+) \geq 1$ whenever $0 \in \{r^-, r^+\}$). Of course each of these functions represents a probability upper-bound, though we entertain values above 1 purely to simplify notation. A persistence or liveness function is *exponential* if it has the form $f(x) = \exp(-\alpha x + \beta)$ for some $\alpha > 0$ and $\beta \geq 0$; ledgers with exponential security will be our main focus.

Finally, our intention is to use dynamic ledgers to model blockchain consensus protocols. In this case, the chain held by each (honest) party $\mathsf{P} \in \mathcal{P}$ is modeled as a dynamic ledger $\mathbf{D}_{\mathsf{P}} = \mathbf{L}_{\mathsf{P}}^{(0)}, \mathbf{L}_{\mathsf{P}}^{(1)}, \ldots$, satisfying the properties of persistence and liveness from Definition 3. Of course, this by itself does not capture all the desired goals of blockchain protocols, as it does not reflect consensus properties across parties; how to reflect this in our model is discussed in the full paper [17].

2.2 Composition of Dynamic Ledgers

In the sequel, we will be interested in combining several dynamic ledgers to form a new "virtual" ledger. This notion of combining makes no assumptions on the ledgers to be combined other than a common transaction space. Moreover, it requires no explicit coordination among the ledgers or maintenance of special metadata: in fact, the "subledgers" involved in the construction do not even need to "know" that they are being viewed as a part of a combined ledger. Concretely, a *virtual ledger construction* is a deterministic, stateless rule for interpreting a family of m individual ledgers as a single ledger. This is formally captured in the following definition.

Definition 4 (Virtual Ledger Constructions). *A* virtual ledger construction $\mathsf{C}[\cdot]$ *is a mapping that takes a tuple of dynamic ledgers* $(\mathbf{D}_1, \ldots, \mathbf{D}_m)$ *over the same transaction space* $\mathcal{T}$ *and returns a dynamic ledger* $\mathsf{C}[\mathbf{D}_1, \ldots, \mathbf{D}_m] = \mathbf{L}^{(0)}, \mathbf{L}^{(1)}, \ldots$ *over* $\mathcal{T}$ *determined by three functions* $(\mathsf{a}_\mathsf{C}, \mathsf{t}_\mathsf{C}, \mathsf{r}_\mathsf{C})$ *as described below. We write* $\mathbf{D}_i = \mathbf{L}_i^{(0)}, \mathbf{L}_i^{(1)}, \ldots$ *with arriving transaction sets denoted* $A_i^{(0)}, A_i^{(1)}, \ldots$ *and the rank function of each* $\mathbf{L}_i^{(t)}$ *being* $\mathsf{rank}_i^{(t)}$*. Then*

(i) the arriving transaction sets are given by $A^{(t)} = \mathsf{a}_{\mathsf{C}}(A_1^{(t)}, \ldots, A_m^{(t)})$*;*
(ii) the ledger contents are given by $\mathbf{L}^{(t)} = \mathsf{t}_{\mathsf{C}}(\mathbf{L}_1^{(t)}, \ldots, \mathbf{L}_m^{(t)})$*; and*
(iii) the rank is given by $\mathsf{rank}^{(t)}(\mathrm{tx}) = \mathsf{r}_{\mathsf{C}}(\mathsf{rank}_1^{(t)}(\mathrm{tx}), \ldots, \mathsf{rank}_m^{(t)}(\mathrm{tx}))$.

Since the above requirements are formulated independently for each t, it is well-defined to treat $\mathsf{C}[\cdot]$ as operating on ledgers rather than dynamic ledgers; we sometimes overload the notation in this sense.

Looking ahead, our amplification combiner will consider $\mathsf{t}_{\mathsf{C}}(\mathbf{L}_1^{(t)}, \ldots, \mathbf{L}_m^{(t)}) = \bigcup_i \mathbf{L}_i^{(t)}$ along with two related definitions of a_{C} given by $\bigcup_i A_i^{(t)}$ and $\bigcap_i A_i^{(t)}$; see Section 3. The robust combiner will adopt a more sophisticated notion of t_{C}.

In each of these cases, the important structural properties of the construction are captured by the rank function r_{C}.

2.3 Transaction Validity and Settlement

In the discussion below, we assume a general notion of *transaction validity* that can be decided inductively: given a ledger $\mathbf{L}$, the validity of a transaction $\mathrm{tx} \in \mathbf{L}$ is determined by the transactions in the state $\mathbf{L}\lceil\mathrm{tx}\rceil$ of $\mathbf{L}$ up to tx and their ordering. Intuitively, only valid transactions are then accounted for when interpreting the state of the ledger on the application level. The canonical example of such a validity predicate in the case of so-called UTXO transactions is formalized in the full version of this paper [17]. Note that protocols such as Bitcoin allow only valid transactions to enter the ledger; as the Bitcoin ledger is represented by a simple chain it is possible to evaluate the validity predicate upon block creation for each included transaction. This may not be the case for more general ledgers, such as the result of applying one of our combiners or various DAG-based constructions.

While we focus our analysis on persistence and liveness as given in Definition 3, our broader goal is to study *settlement*. Intuitively, settlement is the delay necessary to ensure that a transaction included in some $A^{(t)}$ enters the dynamic ledger and, furthermore, that its validity stabilizes for all future times.

Definition 5 (Absolute settlement). *For a dynamic ledger* $\mathbf{D} \stackrel{\text{def}}{=} \mathbf{L}^{(0)}, \mathbf{L}^{(1)}, \ldots$ *we say that a transaction* $\mathrm{tx} \in A^{(\tau)} \cap \mathbf{L}^{(t)}$ *(for $\tau \leq t$) is* (absolutely) settled *at time t if for all $\ell \geq t$ we have: (i)* $\mathbf{L}^{(t)}\lceil\mathrm{tx}\rceil \subseteq \mathbf{L}^{(\ell)}$*, (ii) the linear orders* $\prec_{\mathbf{L}^{(t)}}$ *and* $\prec_{\mathbf{L}^{(\ell)}}$ *agree on* $\mathbf{L}^{(t)}\lceil\mathrm{tx}\rceil$*, and (iii) for any* $\mathrm{tx}' \in \mathbf{L}^{(\ell)}$ *such that* $\mathrm{tx}' \prec_{\mathbf{L}^{(\ell)}} \mathrm{tx}$ *we have* $\mathrm{tx}' \in \mathbf{L}^{(t)}\lceil\mathrm{tx}\rceil$.

Note that for any absolutely settled transaction, its validity is determined and it is guaranteed to remain unchanged in the future.

It will be useful to also consider a weaker notion of *relative settlement* of a transaction: Intuitively, tx is *relatively settled* at time t if we have the guarantee that no (conflicting) transaction tx' that is not part of the ledger at time t can possibly eventually precede tx in the ledger ordering.

Definition 6 (Relative settlement). *Let $\mathcal{T}$ be a transaction space with a conflict relation. For a dynamic ledger $\mathbf{D} \stackrel{\text{def}}{=} \mathbf{L}^{(0)}, \mathbf{L}^{(1)}, \ldots$, over $\mathcal{T}$ we say that a transaction* $\mathrm{tx} \in A^{(\tau)}$ *is* relatively settled *at time $t \geq \tau$ if for any $\ell \geq t$ we have: (i)* $\mathrm{tx} \in \mathbf{L}^{(\ell)}$*; (ii) for any transaction* tx' *such that* $\mathrm{tx}' \prec_{\mathbf{L}^{(\ell)}} \mathrm{tx}$ *and* $\mathrm{tx}' \in \mathrm{conflict}(\mathrm{tx})$ *we have $tx' \in \mathbf{L}^{(t)}$.*

We define an analogous notion when $\mathcal{T}$ is not equipped with a conflict relation, by replacing (ii) with the stronger condition that applies to all transactions: for any transaction tx' *such that $tx' \prec_{\mathbf{L}^{(\ell)}} tx$ we have $tx' \in \mathbf{L}^{(t)}$.*

We illustrate the usefulness of relative settlement on the example of the well-known UTXO transactions. If a UTXO-transaction tx satisfies that: (i) all its inputs appear as outputs of a preceding valid, absolutely settled transaction, (ii) tx itself is relatively settled, and finally, (iii) no conflicting transaction (using the same inputs) is currently part of the ledger; then the validity of tx can be reliably decided and is guaranteed not to change in the future.

In a dynamic ledger with liveness, absolute and relative persistence described by l, p_A and $(\mathrm{p}_R^+, \mathrm{p}_R^-)$ respectively, there is a clear direct relationship of both types of settlement to these properties. Namely, a transaction $\mathrm{tx} \in A^{(\tau)}$ is absolutely (resp. relatively) settled in time $\tau + r_l + r_p$ (resp. $\tau + r_l + r^+ + r^-$) except with error $\mathrm{p}_A(r_p) + \mathrm{l}(r_l)$ (resp. $\mathrm{l}(r_l) + \mathrm{p}_R^+(r^-, r^+) + \mathrm{p}_R^-(r^-)$).

While the time τ when the transaction tx entered the system is not necessarily observable by inspecting the ledger, settlement itself is an observable event: tx is absolutely (resp. relatively) settled at time T if it is seen as part of the ledger $\mathbf{L}^{(T)}\lceil T - r_p \rceil$ (resp. $\mathbf{L}^{(T)}\lceil T - r^+ - r^- \rceil$), except with error $\mathrm{p}_A(r_p)$ (resp. $\mathrm{p}_R^+(r^-, r^+) + \mathrm{p}_R^-(r^-)$).

For ledgers that provide better guarantees for relative persistence than for absolute persistence, relative settlement can occur faster than absolute settlement.

3 The Security-Amplifying Combiner for Latency Reduction

We describe a general combiner which transforms m underlying ledgers to a virtual ledger in which transactions settle more quickly. As discussed previously, by logging a transaction in *all* of the underlying ledgers, users can be promised a $\Theta(m)$ (multiplicative) reduction in settlement time; on the other hand, by logging a transaction in a single one of the underlying ledgers, the promised settlement time is roughly consistent with the underlying ledger settlement time.

3.1 The Subindependence Assumption

Given m dynamic ledgers $\mathcal{D} = (\mathbf{D}_1, \ldots, \mathbf{D}_m)$, informally, we say that the dynamic ledgers satisfy *ε-subindependence* if, for any collection of events $F_1, \ldots, F_m$ capturing either persistence or liveness failures—with the understanding that F_i refers solely to properties of $\mathbf{L}_i$—we have $\Pr\left[\bigwedge_i F_i\right] \leq \prod_i \Pr[F_i]$ conditioned on some event occurring with probability at least $1 - \varepsilon$.

Definition 7 (Subindependence). *Let $\mathcal{D} = (\mathbf{D}_1, \ldots, \mathbf{D}_m)$ be a collection of m dynamic ledgers. Ledgers $\mathcal{D}$ satisfy* ε-persistence subindependence *if for any subset $I \subseteq \{1, \ldots, m\}$ and any collection of persistence failure events $\{P^{(i)}_{r_i,t_i} \mid i \in I\}$, where the event $P^{(i)}_{\star}$ refers to $\mathbf{D}_i$, there is an event E with $\Pr[E] \geq 1 - \varepsilon$ such that we have $\Pr\big[\bigwedge_i P^{(i)}_{r_i,t_i} \;\big|\; E\big] \leq \prod_i \mathrm{p}^{(i)}_A(r_i)$. We similarly define* ε-liveness subindependence.

Throughout our proofs, we treat ε as negligible quantity and, for purposes of a clean exposition, do not include the additive error terms related to ε in our concluding error bounds. (See Section 4.1 for further discussion, including how to interpret the notion of "negligible" in this context.) Consistent with this treatment, we leave ε implicit in our notation, and simply say that the dynamic ledgers $\mathcal{D}$ possess subindependence *if they possess both persistence and liveness subindependence.*

As we discuss in Section 4.1, in situations such as those that arise in blockchains one cannot hope for *exact independence* among persistence failure events for the simple reason that an adaptive adversary may decide—as a result of the success of her attacks on some subset of the ledgers—to cease attacking the others; this creates a (harmless) negative correlation between failure events. Intuitively, the subindependence conditions express the inability of an attacker to outperform the simple setting where she aggressively attacks each of the ledgers in isolation of the others. We discuss how subindependence can be naturally achieved in both PoW and PoS settings in Section 4.1.

3.2 The Parallel Ledger Construction

We consider m dynamic ledgers $\mathcal{D} \stackrel{\text{def}}{=} (\mathbf{D}_1, \ldots, \mathbf{D}_m)$ over the same transaction space $\mathcal{T}$ and sequence of time slots $t \in \{0, 1, \ldots\}$, where each dynamic ledger $\mathbf{D}_i = \mathbf{L}_i^{(0)}, \mathbf{L}_i^{(1)}, \ldots$ and its sequence of arriving transactions is denoted as $A_i^{(0)}, A_i^{(1)}, \ldots$.

Definition 8 (Construction $\mathsf{P}[\mathcal{D}]$). *Our main construction $\mathsf{P}[\mathbf{D}_1, \ldots, \mathbf{D}_m]$ (which we also write $\mathsf{P}[\mathcal{D}]$ when convenient) is defined by*

$$\mathsf{a}_{\mathsf{C}}(A_1^{(t)}, \ldots, A_m^{(t)}) = \bigcup_i A_i^{(t)}\,, \qquad \mathsf{t}_{\mathsf{C}}(\mathbf{L}_1^{(t)}, \ldots, \mathbf{L}_m^{(t)}) = \bigcup_i \mathbf{L}_i^{(t)}\,,$$

and the rank function $\overline{\mathsf{rank}}_L^{(t)}$ defined as follows: For a tuple $\mathbf{r} = (r_1, \ldots, r_m) \in (\mathbb{R} \cup \{\infty\})^m$ and a constant L, define

$$\overline{\mathsf{rank}}_L(\mathbf{r}) \stackrel{\text{def}}{=} -L \ln \left(\frac{1}{m} \sum_{r_i \leq \theta(\mathbf{r})} \exp(-r_i/L) \right), \tag{1}$$

where $\theta(\mathbf{r}) = \min_i r_i + L \ln m$, and $\exp(-\infty/L)$ is defined to be 0. We overload the notation to apply to transactions, so that the resulting rank function can serve the purposes of a virtual ledger construction: Let tx *be a transaction appearing with rank r_i in ledger $\mathbf{L}_i^{(t)}$ for some fixed t; then define $\overline{\mathsf{rank}}_L^{(t)}(\mathrm{tx}) = \overline{\mathsf{rank}}_L(\mathbf{r})$.*

The definition (1) can be rephrased into an alternate, and somewhat more intuitive, equation: if $I_\theta \stackrel{\text{def}}{=} \{i \mid r_i \le \theta(\mathbf{r})\}$ then

$$\frac{1}{m}\sum_{i\in I_\theta} \exp(-r_i/L) = \exp(-\overline{\mathsf{rank}}_L(\mathbf{r})/L)\,. \tag{2}$$

In particular the notion is a simple average if rank is interpreted under an exponential functional: $\exp(-\mathrm{rank}(\cdot)/L)$. Note, additionally, that for any $\mathbf{r} = (r_1, \ldots, r_m)$, we have $\min_{i\in[m]} r_i \le \overline{\mathsf{rank}}_L(\mathbf{r}) \le \big(\min_{i\in[m]} r_i\big) + L \ln m$ and, furthermore, the inequality can be naturally interpreted if some or all of the r_i are ∞. The first inequality is tight when all r_i are equal.

A final remark about truncation by the threshold $\theta(\mathbf{r})$: While the "large-scale" features of the parallel ledger—including relative persistence and liveness—do not depend on truncation, absolute persistence depends on eventual stability of the rank function. The truncation operation guarantees this, ensuring that only a bounded portion of the ledger is relevant for determining the final rank of a transaction.

Preemptive rank function. When the dynamic ledgers $\mathcal{D}$ are defined over a transaction space with a conflict relation, we consistently work with a slightly different notion of *preemptive rank* for the amplification construction above. Specifically, we say that a transaction tx is *dominant* in a ledger $\mathbf{L}$ if it appears in the ledger and no earlier transaction conflicts with tx (that is $\mathrm{tx} \in \mathbf{L}$ and $\mathrm{tx}' \prec_{\mathbf{L}} \mathrm{tx} \Rightarrow \mathrm{tx}' \notin \mathrm{conflict}(\mathrm{tx})$). Let ρ_i be the rank of tx in ledger $\mathbf{L}_i$ and define $r_i = \rho_i$ if tx is dominant in $\mathbf{L}_i$, and $r_i = \infty$ otherwise. Then the *preemptive rank* $\overline{\mathsf{rank}}^*_L(\mathrm{tx})$ of tx is defined to be $\overline{\mathsf{rank}}_L(r_1, \ldots, r_m)$.

Fast and slow submission. We consider two ways of submitting tx to $\mathsf{P}[\mathcal{D}]$:

The "fast" mechanism: A transaction tx is simultaneously submitted to all of the underlying dynamic ledgers $\{\mathbf{D}_i\}_{i=1}^m$, appearing in $\bigcap_{i\in[m]} A_i^{(t)}$.

The "slow" mechanism: A transaction tx is submitted to (at least) one of the dynamic ledgers $\mathbf{D}_i$, appearing in $\bigcup_{i\in[m]} A_i^{(t)}$.

An important feature of our protocol is that a single deployment supports both of these mechanisms and their use can be decided by transaction producers on a per-transaction basis. As we will see, these two mechanisms exhibit markedly different liveness guarantees: Participants desiring fast liveness and settlement[5] can adopt the fast mechanism by submitting their transactions to all m of the ledgers; participants with less urgency can adopt the slow mechanism, simply submitting their transactions to a single ledger.

To formally capture this in a clean way, we will introduce a slight variant, $\mathsf{P}_F[\mathcal{D}]$, which allows us to specifically study the improved liveness properties of transactions when they happen to be submitted for insertion into all of the

[5] Recall the difference between *liveness* and *settlement* in our terminology (cf. Section 2.3).

constituent ledgers $\mathbf{D}_i$ at the same time. Specifically, $\mathsf{P}_F[\mathcal{D}]$ has precisely the same definition as $\mathsf{P}[\mathcal{D}]$ with the exception that $\mathsf{a_C}(A_1^{(t)}, \ldots, A_m^{(t)}) = \bigcap_i A_i^{(t)}$. Thus, note that the two virtual ledgers $\mathsf{P}[\mathcal{D}]$ and $\mathsf{P}_F[\mathcal{D}]$ contain exactly the same elements with exactly the same ranks. They differ only in the sets of transactions (determined by $\mathsf{a_C}$) for which they provide liveness guarantees: "slow" liveness guarantees for $\bigcup_i A_i^{(t)}$ correspond to bounds on $\mathsf{P}[\mathcal{D}]$ while "fast" liveness guarantees for transactions in $\bigcap_i A_i^{(t)}$ correspond to liveness guarantees for $\mathsf{P}_F[\mathcal{D}]$. This bookkeeping slight of hand is merely a way to use a single abstraction to express both a general liveness guarantee and an accelerated guarantee for transaction submitted to all ledgers $\mathbf{D}_i$.

We remark that fast settlement guarantees are provided anytime a transaction has been submitted to all of the underlying ledgers: the proof does not require that they be submitted at exactly the same time. In terms of the definitions above, the proof would apply even if we defined $\overline{A}_i^{(t)} \stackrel{\text{def}}{=} \bigcup_{s \le t} A_i^{(s)}$, $F^{(t)} \stackrel{\text{def}}{=} \bigcap_i \overline{A}_i^{(t)}$, and $A^{(t)}$ (the set for which fast settlement is guaranteed) to be $F^{(t)} \setminus F^{(t-1)}$. Thus a transaction would be guaranteed fast settlement as soon as it has been submitted to all relevant ledgers. We work with the simple formulation ($\bigcap_i A_i^{(t)}$) merely as a matter of convenience.

3.3 Main Result and Proof Outline

Our main result follows, formulated for exponentially secure ledgers as defined in Section 2.1.

Theorem 2. *Let $\mathcal{D} = (\mathbf{D}_1, \ldots, \mathbf{D}_m)$ be a family of m subindependent dynamic ledgers defined over a common transaction space $\mathcal{T}$ with a conflict relation, each possessing exponential liveness $\mathrm{l}(r) = \exp(-\alpha_\mathrm{l} r + \beta_\mathrm{l})$ and absolute persistence $\mathrm{p}(r) = \exp(-\alpha_\mathrm{p} r + \beta_\mathrm{p})$. Consider the combined dynamic ledgers $\mathsf{P}_F[\mathcal{D}]$ and $\mathsf{P}[\mathcal{D}]$ with the (preemptive) rank function $\overline{\mathsf{rank}}_L^*$ for a parameter $L \ge m$. Then for $\mathsf{P}_F[\mathcal{D}]$, there is a constant $C > 1$ so that if $L \ge Cm \ln m$, we have*

$$\begin{aligned} &\Pr\left[\exists \mathrm{tx} \in A^{(t_0)} \text{ not relatively settled at time } t_0 + 2r\right] \\ &\quad \le \exp(-r\Omega(m) + O(m)) + \exp\left(-\Omega(r) - \Omega(L \ln(m))\right) . \end{aligned} \tag{3}$$

At the same time, for $\mathsf{P}[\mathcal{D}]$ we have

$$\Pr\left[\exists \mathrm{tx} \in A^{(t_0)} \text{ not absolutely settled at } t_0 + 2r\right] \le m \exp(-\Omega(r) + O(L \ln m)) .$$

The constants hidden in the $\Omega()$ and $O()$ notation depend on $\alpha_\mathrm{p}, \alpha_\mathrm{l}, \beta_\mathrm{p}, \beta_\mathrm{l}$, but they are independent of m, L, and r.

Note that in (3), the first term vanishes with the desired m-fold speedup, and dominates the total error as long as roughly $rm < L$. Beyond that, the second term is dominant and the error vanishes at the pace of a single constituent ledger. This is essential for enabling both slow and fast settlement, as discussed

in Section 1.1. Note that as L can be chosen to scale with the security parameter so that $\exp(-\Theta(L))$ is an acceptable error probability, the region $rm < L$ is thus exactly where the settlement speedup is desired.

On a high level, the proof for $\mathsf{P}_F[\mathcal{D}]$ goes as follows. For a transaction $\mathrm{tx} \in A^{(t_0)}$, we can expect that: (1) At time $t_0 + 2r$, tx appears in at least $4m/5$ of the m ledgers with rank at most $t_0 + r$. (2) At most $m/5$ of these $4m/5$ ledgers will exhibit an absolute persistence failure allowing a change of their state up to rank $t_0 + r$ after time $t_0 + 2r$, affecting the rank of tx. Based on the above two events, at any time after $t_0 + 2r$ there can be at most $2m/5$ ledgers that do not contain tx with rank at most $t_0 + r$. Then: (3) For any competing transaction $\mathrm{tx}' \in \mathrm{conflict}(\mathrm{tx})$ not present at time $t_0 + 2r$, these $2m/5$ ledgers will never contribute enough to the rank of tx' to overtake tx in $\mathsf{P}_F[\mathcal{D}]$. More precisely, each of the three above events is shown to fail with at most the error probability in the theorem statement.

The result for $\mathsf{P}[\mathcal{D}]$ is proven along the following lines. Assume a transaction tx inserted to (at least) one of the ledgers $\mathbf{L}_i$ at time t_0. For any $t \geq t_0 + r - L \ln m$, we have $\mathrm{tx} \in \mathbf{L}_i^{(t)}\lceil t_0 + r - L \ln m \rceil$ except for probability $\mathrm{l}(r - L \ln m)$, and, by the properties of the rank function, also $\mathrm{tx} \in \mathbf{L}^{(t)}\lceil t_0 + r \rceil$. Let $T \geq t = t_0 + 2r$ and assume $\mathrm{tx} \in \mathbf{L}^{(t)}\lceil t_0 + r \rceil$ as by the above liveness guarantee. As $\mathbf{L}^{(T)}\lceil t_0 + r \rceil$ is fully determined by the ledgers $\mathbf{L}_j^{(T)}\lceil t_0 + r + L \ln m \rceil$, a persistence failure $\mathbf{L}^{(T)}\lceil t_0 + r \rceil \neq \mathbf{L}^{(t)}\lceil t_0 + r \rceil$ implies a persistence failure of some $\mathbf{L}_j^{(t)}\lceil t_0 + r + L \ln m \rceil$, which has a probability at most $m\,\mathrm{p}_A(-r + L \ln m)$.

The bound for $\mathsf{P}_F[\mathcal{D}]$ in particular gives us the following corollary.

Corollary 1. *In the setting of Theorem 2, if the number of chains m scales with the security parameter then $\mathsf{P}_F[\mathcal{D}]$ achieves constant-time settlement except with an error probability negligible in the security parameter.*

In the rest of this section, we establish the above results in full detail. In Section 3.4 we study the central part of our combiner—its rank function; and based on it, Section 3.5 obtains our persistence and liveness bounds in their most general form. Section 3.6 specializes them to the setting of interest with exponentially-secure underlying ledgers; and finally Section 3.7 concludes the derivation of Theorem 2 and Corollary 1.

3.4 Properties of $\overline{\mathsf{rank}}$

Before discussing the persistence and liveness guarantees of our construction, we derive some general properties of its rank function.

Lemma 1. *Let $\mathbf{r} = (r_1, \ldots, r_m) \in (\mathbb{R} \cup \{\infty\})^m$ and $T \geq \min_i r_i$. Let $I_T = \{i \mid r_i \leq T\}$ and, for each $i \in I_T$, define $d_i = T - r_i$. Writing $D = T - \overline{\mathsf{rank}}_L(\mathbf{r})$,*

$$\sum_{i \in I_T} d_i \geq D + L \ln \left(m - \frac{m-1}{\exp(D/L)} \right) .$$

We note the following weaker, but convenient, bound: when $D \geq 0$, the sum $\sum_{i \in I_T} d_i$ is no more than $D + L \ln ((mD + L)/(D + L))$.

Proof. Let $I_\theta = \{i \mid r_i \le \theta(\mathbf{r})\}$. Writing $R = -\overline{\mathsf{rank}}_L(\mathbf{r})/L$, from equation (2) we have

$$m\exp(T/L)\exp(R) = \exp(T/L)\sum_{i\in I_\theta}\exp(-r_i/L) \le \sum_{i\in I_\theta\setminus I_T} 1 + \sum_{i\in I_T\cap I_\theta}\exp(d_i/L)$$
$$\overset{(*)}{\le} (|I_\theta|-1) + \exp\left(\sum_{i\in I_T} d_i/L\right) \le (m-1) + \exp\left(\sum_{i\in I_T} d_i/L\right) \tag{4}$$

where the inequality $\overset{(*)}{\le}$ above follows from the fact that for any $a_i \ge 0$ we have $\sum_{i=1}^{\ell}\exp(a_i) \le (\ell-1) + \exp(\sum_{i=1}^{\ell} a_i)$, and $d_i \ge 0$ for all $i \in I_T$. (This follows by expanding the power series of e^x and noting that $\sum a_i^k \le (\sum a_i)^k$ for positive a_i.) Inequality (4) then yields

$$\sum_{i\in I_T}\frac{d_i}{L} \ge \ln\left[m\exp\left[\frac{T}{L}+R\right]-(m-1)\right] = \left[\frac{T}{L}+R\right] + \ln\left[m - \frac{m-1}{\exp(T/L+R)}\right]$$

and hence

$$\sum_{i\in I_T} d_i \ge (T - \overline{\mathsf{rank}}_L(\mathbf{r})) + L\ln\left(m - \frac{m-1}{\exp([T-\overline{\mathsf{rank}}_L(\mathbf{r})]/L)}\right),$$

completing the proof. The second lower bound indicated in the theorem follows from the fact that $\exp(1+x) \ge 1+x$ for $x \ge 0$. □

We note a corollary of this, which also reflects the number of contributing terms in the sum defining $\overline{\mathsf{rank}}$.

Corollary 2. *Let* $\mathbf{r} = (r_1,\dots,r_m) \in (\mathbb{R}\cup\{\infty\})^m$ *and* $T \ge \min_i r_i$. *Let*

$$I_T = \{i \mid r_i \le T\}, \qquad I_\theta = \{i \mid r_i \le \theta(\mathbf{r})\}, \qquad m' = |I_\theta|,$$

and, for each $i \in I_T$, *define* $d_i = T - r_i$. *Then*

$$\sum_{i\in I_T} d_i \ge \left[T - \overline{\mathsf{rank}}_L(\mathbf{r})\right] + L\ln\left(m - \frac{m'-1}{\exp([T-\overline{\mathsf{rank}}_L(\mathbf{r})]/L)}\right).$$

Proof. This follows from the proof of Lemma 1 by working with the version of Equation (4) that retains dependence on $|I_\theta|$. □

For two rank tuples $\mathbf{r} = (r_1,\dots,r_m)$ and $\mathbf{s} = (s_1,\dots,s_m)$ in $(\mathbb{R}\cup\{\infty\})^m$, we define $\mathbf{r}\vee\mathbf{s}$ to be the tuple $(\min(r_1,s_1),\dots,\min(r_m,s_m))$.

Lemma 2 (Rank addition). *Consider two rank tuples* $\mathbf{r} = (r_1,\dots,r_m)$ *and* $\mathbf{s} = (s_1,\dots,s_m)$ *in* $(\mathbb{R}\cup\{\infty\})^m$. *Then*

$$\exp(-\overline{\mathsf{rank}}(\mathbf{r}\vee\mathbf{s})/L) \le \exp(-\overline{\mathsf{rank}}(\mathbf{r})/L) + \exp(-\overline{\mathsf{rank}}(\mathbf{s})/L); \tag{5}$$

and moreover, for any $\alpha \in (0,1)$,

$$\overline{\mathsf{rank}}(\mathbf{r}) \ge \overline{\mathsf{rank}}(\mathbf{r}\vee\mathbf{s}) + \ln(1/\alpha)L \implies \overline{\mathsf{rank}}(\mathbf{s}) \le \overline{\mathsf{rank}}(\mathbf{r}\vee\mathbf{s}) + \ln(1/(1-\alpha))L. \tag{6}$$

Proof. The validity of equation (5) can be observed by simply expanding the $\overline{\mathsf{rank}}$ function according to its definition. For the implication (6), note that if $\overline{\mathsf{rank}}(\mathbf{r}) \geq \overline{\mathsf{rank}}(\mathbf{r} \vee \mathbf{s}) + \ln(1/\alpha)L$ then (5) gives us

$$\exp\left(-\overline{\mathsf{rank}}(\mathbf{r} \vee \mathbf{s})/L\right) \leq \alpha \cdot \exp\left(-\overline{\mathsf{rank}}(\mathbf{r} \vee \mathbf{s})/L\right) + \exp\left(-\overline{\mathsf{rank}}(\mathbf{s})/L\right)$$

and hence $\exp\left(-\overline{\mathsf{rank}}(\mathbf{s})/L\right) \geq (1-\alpha)\cdot\exp\left(-\overline{\mathsf{rank}}(\mathbf{r} \vee \mathbf{s})/L\right)$, implying $\overline{\mathsf{rank}}(\mathbf{s}) \leq \overline{\mathsf{rank}}(\mathbf{r} \vee \mathbf{s}) + \ln(1/(1-\alpha))L$ as desired. □

3.5 Persistence and Liveness of the Parallel Ledgers

We begin with a lemma that establishes relative persistence guarantees under general circumstances: it requires only a super-additive persistence function and does not require that the transaction space have a conflict relation.

Definition 9 (Super-additive functions). *Recall that a function $f : \mathbb{R} \to \mathbb{R}$ is convex if, for any $x_1, \ldots, x_n$ and $\lambda_1, \ldots, \lambda_n$ for which $\lambda_i \geq 0$ and $\sum_i \lambda_i = 1$, we have $f(\sum_i \lambda_i x_i) \leq \sum_i \lambda_i f(x_i)$. A persistence function* p *is* super-additive *if* $\log \mathrm{p}$ *is convex. It follows that* p *satisfies the inequality*

$$\prod_{i=1}^{m} \mathrm{p}(r_i) \leq \mathrm{p}\left(\frac{1}{m}\sum_i r_i\right)^m . \tag{7}$$

Note that any exponential persistence function (as defined in Sect. 2.1) is super-additive.

Lemma 3 (Relative persistence of $\mathsf{P}[\mathcal{D}]$). *Consider $\mathsf{P}[\mathcal{D}]$, the parallel composition of m subindependent ledgers, each with super-additive absolute persistence $\mathrm{p}_A(\cdot)$. For any $\delta > 0$ and time T, the probability that an adversary can inject a transaction* tx *that does not appear in any of the ledgers so as to achieve $\overline{\mathsf{rank}}_L(\mathrm{tx}) \leq T - D$ is at most*

$$i(D; \delta, L) \stackrel{\text{def}}{=} \left(\frac{D + L\ln m}{\delta}\right)^m \cdot \mathrm{p}_A\left(\frac{1}{m}\left(D + L\ln\left(\frac{mD+L}{D+L}\right)\right) - \delta\right)^m .$$

Moreover, the ledger $\mathsf{P}[\mathcal{D}]$ satisfies the following relative persistence guarantees: for any $t_0, r \geq 0$,

$$\Pr\left[\exists t \geq t_0, \mathbf{L}^{(t)}\left\lceil t_0 - r\right\rceil \not\subseteq \mathbf{L}^{(t_0)}\right] \leq \overline{\mathrm{p}}^{-}_R(r; L) \stackrel{\text{def}}{=} i(r; \delta, L)$$

and, for the constant $r^* = \ln(2)L$,

$$\Pr\left[\exists t \geq t_0, \mathbf{L}^{(t_0)}\left\lceil t_0 - (r + r^*)\right\rceil \not\subseteq \mathbf{L}^{(t)}\left\lceil t_0 - r\right\rceil\right] \leq \overline{\mathrm{p}}^{+}_R(r, r^*; L) \stackrel{\text{def}}{=} i(r; \delta, L)\,.$$

Proof. In light of Lemma 1, in order for a transaction tx to be injected into the m ledgers so as to achieve $\overline{\mathsf{rank}}_L(\mathrm{tx}) \leq T - D$, it must appear with a rank tuple $(T - d_1, \ldots, T - d_m)$ for which

$$\sum_i d_i \geq D + L\ln\left(\frac{mD+L}{D+L}\right) .$$

In preparation for applying a union bound, we identify a finite family of tuples $\mathcal{R}$ so that for any tuple of positive reals $\mathbf{x} = (x_1, \ldots, x_m)$ with $\sum x_i \geq \Lambda$ there is a "bounding" tuple $\mathbf{r} \in \mathcal{R}$ so that $\mathbf{r} \leq \mathbf{x}$ and $\sum_i r_i \approx \Lambda$. (Here the $\leq$ indicates that $r_i \leq x_i$ for all i.) For two real numbers x and $\delta > 0$, define $\lfloor x \rfloor_\delta$ to be the largest integer multiple of δ that is less than or equal to x; that is, $\lfloor x \rfloor_\delta \stackrel{\text{def}}{=} \max\{k \in \delta\mathbb{Z} \mid k \leq x\}$. Observe that for any tuple $\mathbf{x} = (x_1, \ldots, x_m)$ for which $\sum_i x_i \geq \Lambda$, the tuple $\lfloor \mathbf{x} \rfloor_\delta \stackrel{\text{def}}{=} (\lfloor x_1 \rfloor_\delta, \ldots, \lfloor x_m \rfloor_\delta)$ contains only integer multiples of δ, is coordinate-wise no larger than $\mathbf{x}$, and satisfies $\Lambda - \delta m \leq \sum_i \lfloor x_i \rfloor_\delta \leq \Lambda$. For $\Lambda \geq 0$, let $\mathcal{R}(\Lambda, \delta) = \{\mathbf{r} = (r_1, \ldots, r_m) \mid r_i \in \delta\mathbb{Z}, r_i \geq 0, \Lambda - \delta m \leq \sum_i r_i \leq \Lambda\}$. With this in place, it follows that if tx appears with ranks $(T - d_1, \ldots, T - d_m)$ and $T - \overline{\mathsf{rank}}_L(\mathrm{tx}) \geq D$ then there is a tuple

$$\mathbf{r} \in \mathcal{R} \stackrel{\text{def}}{=} \mathcal{R}\left(D + L \ln\left[\frac{mD + L}{D + L}\right], \delta\right)$$

for which $\mathbf{r} \leq \mathbf{d}$ and hence $(T - d_1, \ldots, T - d_m) \leq (T - r_1, \ldots, T - r_m)$.

For a tuple $\mathbf{r} = (r_1, \ldots, r_m)$ consider the event, denoted $E_{\mathbf{r}}$, that the adversary can inject a transaction so that it appears with rank no more than $T - r_i$ in ledger i. By subindependence and the convexity of $\log \mathrm{p}_A(\cdot)$,

$$\Pr[E_{\mathbf{r}}] \leq \prod_{i=1}^{m} \mathrm{p}_A(r_i) \leq \mathrm{p}_A\left(\frac{1}{m}\sum_{i=1}^{m} r_i\right)^m,$$

from inequality (7) above. Then we have

$$\Pr\left[\text{tx injected so that } \overline{\mathsf{rank}}_L(\mathrm{tx}) \leq T - D\right] \leq |\mathcal{R}| \cdot \max_{\mathbf{r} \in \mathcal{R}} \Pr[E_{\mathbf{r}}].$$

To conclude the argument, invoking the upper bound $|\mathcal{R}| \leq ((D + L \ln m)/\delta)^m$ we see that the probability $\Pr[\text{tx injected so that } \overline{\mathsf{rank}}_L(\mathrm{tx}) \leq T - D]$ is bounded above by

$$\left(\frac{D + L \ln m}{\delta}\right)^m \cdot \mathrm{p}_A\left(\frac{1}{m}\left(D + L \ln\left[\frac{mD + L}{D + L}\right]\right) - \delta\right)^m.$$

The bound on $\mathrm{p}_R^-(r)$ follows immediately.

As for $\mathrm{p}_R^+(r, \ln(2)L; L)$, consider a transaction tx with rank $T - (r + \ln(2)L)$. In order for such a transaction to rise to rank $T - r$, some subset S of appearances of the transaction must be removed with sufficient rank to permit the resulting rank to rise to $T - r$. In light of Lemma 2, this removal must involve rewriting the underlying blockchains at ranks corresponding to $\overline{\mathsf{rank}}$ at least $T - (r + \ln(2)L) + \ln(2)L = T - r$, as desired. (This corresponds to the setting $\alpha = 1/2$ in Lemma 2). □

We state a corollary of the above result which pertains to the problem of injecting a transaction into a *particular* subset of the ledgers. This relies directly on Corollary 2, and will be a critical component of the $\Theta(m)$-amplification results below.

Corollary 3 (Relative persistence of $\mathsf{P}[\mathcal{D}]$ with targeted insertion). *Consider $\mathsf{P}[\mathcal{D}]$, the parallel composition of m subindependent ledgers, each with superadditive absolute persistence $\mathrm{p}_A(\cdot)$. Let $\mathcal{I}$ denote a subset of m' of the ledgers and let D satisfy $\exp(D/L) > (m'-1)/(m-1)$. Then for any $\delta > 0$ and time T, the probability that an adversary can inject a transaction tx that does not appear in any of the ledgers so as to appear only in ledgers $\mathcal{I}$ and achieve $\overline{\mathsf{rank}}_L(\mathrm{tx}) \leq T - D$ is no more than*

$$i(D, m'; \delta, L) \stackrel{\text{def}}{=} \left(\frac{D + L\ln m}{\delta}\right)^{m'} \cdot \mathrm{p}_A\left(\frac{1}{m'}\left(D + L\ln\left(m - \frac{m'-1}{\exp(D/L)}\right)\right)\right) - \delta\Bigg)^{m'}.$$

Proof. This follows from the proof of Lemma 3 by suitably adjusting the bound on $|\mathcal{R}|$ to the restricted set of chains and applying the bound from Corollary 2. □

We return to the general setting to formulate a bound on absolute persistence.

Lemma 4 (Absolute persistence of $\mathsf{P}[\mathcal{D}]$). *Consider $\mathsf{P}[\mathcal{D}]$, the parallel composition of m subindependent ledgers, each with absolute persistence $\mathrm{p}_A(\cdot)$. Then the parallel ledger $\mathsf{P}[\mathcal{D}]$ has absolute persistence $\overline{\mathrm{p}}_A(r) \leq m\,\mathrm{p}_A(r - L\ln m)$.*

Proof. As above, we let $\mathsf{P}[\mathbf{D}_1, \ldots, \mathbf{D}_m] = \mathbf{L}^{(0)}, \mathbf{L}^{(1)}, \ldots$. Consider a time t_0 and $r \geq L\ln m$. We observe that for any time $t \geq t_0$, $\mathbf{L}^{(t)}\lceil t_0 - r\rceil$ is completely determined by the ledgers $\mathbf{L}_i^{(t)}\lceil t_0 - r + L\ln m\rceil$. To see this, consider a transaction tx in the general ledger $\mathbf{L}^{(t)}$ of rank $s \leq t_0 - r$. Letting s_i denote the rank of tx in the constituent ledgers $\mathbf{L}_i^{(t)}$, recall that $\min_i s_i \leq s \leq t_0 - r$ and, furthermore, that $s = \overline{\mathsf{rank}}(\mathrm{tx})$ depends only on those s_i for which

$$s_i \leq \theta(\mathbf{s}) = \min_i s_i + L\ln m \leq s + L\ln m \leq t_0 - r + L\ln m\,;$$

in particular $\overline{\mathsf{rank}}(\mathrm{tx})$ is determined only by the ledgers $\mathbf{L}_i^{(t)}\lceil t_0 - r + L\ln m\rceil$.

To conclude, a persistence failure in $\mathbf{L}^{(t)}\lceil t_0 - r\rceil$ implies a persistence failure in some $\mathbf{L}_i^{(t)}\lceil t_0 - r + L\ln m\rceil$ and thus $\overline{\mathrm{p}}_A(r) \leq m\,\mathrm{p}_A(r - L\ln m)$, as desired. □

As the ledger $\mathsf{P}_F[\mathcal{D}]$ is identical to $\mathsf{P}[\mathcal{D}]$ aside from the definition of a_C, it possesses the persistence guarantees described in Lemma 3, Corollary 3, and Lemma 4.

Liveness. We now direct our attention to liveness. We separately consider two distinct ways of submitting a transaction to the parallel ledger, the "fast" and the "slow" mechanisms as defined in Section 3.2. Recall that formally, the "fast" case corresponds to the liveness function of the virtual ledger $\mathsf{P}_F[\mathcal{D}]$, while the "slow" case corresponds to the liveness of the virtual ledger $\mathsf{P}[\mathcal{D}]$. We study these liveness functions next.

Definition 10 (Census). *Consider $\mathsf{P}[\mathbf{D}]$, and let $\mathrm{tx} \in \mathcal{T}$ be a transaction. The (r, T)-*census *of tx, denoted by $C_r^{(T)}(\mathrm{tx})$, is the number of ledgers for which $\mathrm{tx} \in \mathbf{L}_i^{(T)}\lceil r\rceil$. When T can be inferred from context, we shorten this to the r-census $C_r(\mathrm{tx})$.*

Lemma 5 (Liveness of $\mathsf{P}_F[\mathcal{D}]$). *Consider $\mathsf{P}_F[\mathcal{D}]$, the parallel composition of m subindependent ledgers, each with liveness $\mathrm{l}(\cdot)$. Then, for any t_0 and t for which $t \geq t_0 + r$ and any $\gamma \in [0,1]$,*

$$\Pr[\exists \mathrm{tx} \in \bigcap A_i^{(t_0)} \text{ with } (t_0+r,t)\text{-census} \leq (1-\gamma)m] \leq \binom{m}{\gamma m} \mathrm{l}(r)^{\gamma m} .$$

It follows that for any $\gamma \in (0,1)$ the ledger $\mathsf{P}_F[\mathcal{D}]$ has liveness

$$\bar{\mathrm{l}}^{\mathsf{P}_F}(r) = \binom{m}{\gamma m} \mathrm{l}\left(r - L \ln\left(\frac{1}{1-\gamma}\right)\right)^{m\gamma} .$$

Proof. Consider times $t \geq t_0$ and a delay $r \geq 0$. For a parameter $\gamma \in (0,1)$ we consider the (census) event that the transactions in $\bigcap_i A_i^{(t_0)}$ appear in at least $(1-\gamma)m$ of the ledgers $\mathbf{L}_i^{(t)}\lceil t_0 + r\rceil$. In this case, any transaction $\mathrm{tx} \in A^{t_0}$ has rank $\overline{\mathsf{rank}}(\mathrm{tx}) \leq t_0 + r + L\ln(1/(1-\gamma))$ in the ledger $\mathbf{L}^{(t)}$. It follows that the probability that there exists a transaction in $A^{(t_0)}$ that does not appear in $\mathbf{L}^{(t)}\lceil t_0 + r + L\ln(1/(1-\gamma))\rceil$ is no more than $\binom{m}{\gamma m}\mathrm{l}(r)^{\gamma m}$. Reparametrizing this (by setting $r' = r + L\ln(1/\gamma)$) yields the statement of the lemma. □

Lemma 6 (Liveness of $\mathsf{P}[\mathcal{D}]$). *Consider $\mathsf{P}[\mathcal{D}]$, the parallel composition of m ledgers, each with liveness $\mathrm{l}(\cdot)$. Then the parallel ledger $\mathsf{P}[\mathcal{D}]$ has liveness $\bar{\mathrm{l}}^{\mathsf{P}}(r) = \mathrm{l}(r - L\ln m)$.*

Proof. Consider times $t \geq t_0$ and a delay $r \geq 0$. Observe that if a transaction tx appears in any $\mathbf{L}_i^{(t)}\lceil t_0 + r\rceil$ then it appears in $\mathbf{L}^{(t)}\lceil t_0 + r + L\ln m\rceil$. This yields the statement of the lemma. □

3.6 Ledgers with Exponential Security

To achieve guarantees with more immediate interpretability and prepare for our main amplification results, we consider the most interesting case for persistence and liveness functions: $r \mapsto \exp(-\alpha r + \beta)$ for $\alpha, \beta \geq 0$. Note that such a function is superadditive according to Definition 9. The following statements follow directly from Corollary 3 with $\delta = 1$, and from Lemmas 4–6.

Corollary 4 (Relative persistence with targeted insertion). *Consider $\mathsf{P}[\mathcal{D}]$ or $\mathsf{P}_F[\mathcal{D}]$, the parallel composition of m ledgers, each with absolute persistence $\mathrm{p}_A(r) = \exp(-\alpha_{\mathrm{p}} r + \beta_{\mathrm{p}})$. Let $\mathcal{I}$ denote a subset of m' of the ledgers and let D satisfy $\exp(D/L) > (m'-1)/(m-1)$. Then for any $\delta > 0$ and time T, the probability that an adversary can inject a transaction tx that does not appear in any of the ledgers so as to appear only in ledgers $\mathcal{I}$ and achieve $\overline{\mathsf{rank}}_L(\mathrm{tx}) \leq T - D$ is no more than*

$$(D + L\ln m)^{m'} \cdot \exp\left(-\alpha_{\mathrm{p}}\left[D + L\ln\left(m - \frac{m'-1}{\exp(D/L)}\right)\right] + (\alpha_{\mathrm{p}} + \beta_{\mathrm{p}})m'\right) .$$

Corollary 5 (Absolute persistence). *Consider* $\mathsf{P}[\mathcal{D}]$ *or* $\mathsf{P}_F[\mathcal{D}]$*, the parallel composition of* m *ledgers with absolute persistence* $\mathrm{p}_A(r) = \exp(-\alpha_{\mathrm{p}} r + \beta_{\mathrm{p}})$*. Then the ledgers* $\mathsf{P}[\mathcal{D}]$ *and* $\mathsf{P}_F[\mathcal{D}]$ *both have absolute persistence* $\overline{\mathrm{p}}_A(r) \leq m^{\alpha_{\mathrm{p}} L+1} \exp(-\alpha_{\mathrm{p}} r + \beta_{\mathrm{p}})$*.*

Corollary 6 (Liveness). *Consider* $\mathsf{P}[\mathcal{D}]$ *and* $\mathsf{P}_F[\mathcal{D}]$*, constructed with* m *ledgers* $\mathcal{D}$ *that each possess liveness* $\mathrm{l}(r) = \exp(-\alpha_{\mathrm{l}} r + \beta_{\mathrm{l}})$*. Then, for any* $\gamma \in (0,1)$ *and times* t_0 *and* t *for which* $t_0 + r \leq t$*,*

$$\Pr[\exists \mathrm{tx} \in A^{(t_0)} \text{ with } (t_0 + r, t)\text{-census} \leq (1-\gamma)m] \leq \binom{m}{\gamma m} \exp(-\gamma m(\alpha_{\mathrm{l}} r - \beta_{\mathrm{l}}))$$

and the liveness function $\bar{\mathrm{l}}^{\mathsf{P}_F}(\cdot)$ *of* $\mathsf{P}_F[\mathcal{D}]$ *satisfies*

$$\bar{\mathrm{l}}^{\mathsf{P}_F}(r) = \binom{m}{\gamma m} \exp\left(-\alpha_{\mathrm{l}} \gamma m \left(r - L \ln\left(\frac{1}{1-\gamma}\right)\right) + \beta_{\mathrm{l}} \gamma m\right).$$

The liveness function $\bar{\mathrm{l}}^{\mathsf{P}}(\cdot)$ *of* $\mathsf{P}[\mathcal{D}]$ *satisfies* $\bar{\mathrm{l}}^{\mathsf{P}}(r) = m^{\alpha L} \exp(-\alpha_{\mathrm{l}} r + \beta_{\mathrm{l}})$*.*

Theorem 3 (Restatement of Theorem 2 for $\mathsf{P}[\mathcal{D}]$**).** *Consider* $\mathsf{P}[\mathcal{D}]$ *for a family of* m *subindependent ledgers* $\mathcal{D} = (\mathbf{D}_1, \ldots, \mathbf{D}_m)$*, each possessing exponential liveness* $\mathrm{l}(r) = \exp(-\alpha_{\mathrm{l}} r + \beta_{\mathrm{l}})$ *and (absolute) persistence* $\mathrm{p}(r) = \exp(-\alpha_{\mathrm{p}} r + \beta_{\mathrm{p}})$*. We assume all ledgers are defined over a common transaction space* $\mathcal{T}$ *with a conflict relation and the general ledger is defined over the (preemptive) rank function* $\overline{\mathsf{rank}}^*_L$ *for a parameter* $L \geq m$*. Then*

$$\Pr\begin{bmatrix}\exists \mathrm{tx} \in A^{(t_0)} \text{ not absolutely} \\ \text{settled at time } t_0 + 2r\end{bmatrix} \leq m \exp(-\Omega(r) + O(L \ln m)).$$

The constants hidden in the $\Omega()$ *and* $O()$ *notation depend on* $\alpha_{\mathrm{p}}, \alpha_{\mathrm{l}}, \beta_{\mathrm{p}}, \beta_{\mathrm{l}}$*, but they are independent of* m*,* L*, and* r*.*

Proof. Assume a transaction tx inserted to (at least) one of the ledgers $\mathbf{L}_i$ at time t_0. By Corollary 6, at any point in time $t \geq t_0 + r$, we have that $\mathrm{tx} \in \mathbf{L}^{(t)}\lceil t_0 + r \rceil$ except for probability $\bar{\mathrm{l}}(r) \leq \exp(-\Omega(r) + O(L \ln m))$. Let $T \geq t = t_0 + 2r$. By Corollary 5, $\mathbf{L}^{(T)}\lceil t_0 + r \rceil = \mathbf{L}^{(t)}\lceil t_0 + r \rceil$ remains persistent except for error $\overline{\mathrm{p}_A}(r) \leq m \exp(-\Omega(r) + O(L \ln m))$. The stated bound now follows by union bound over the errors $\bar{\mathrm{l}}(r)$ and $\overline{\mathrm{p}_A}(r)$. □

3.7 Fast Settlement with Preemption: Achieving Linear Amplification and Constant Settlement Time

We show how to achieve $\Theta(m)$ amplification for liveness and settlement time. This construction applies to transaction spaces with a conflict relation, and focuses on the setting of ledgers with exponential security, as discussed in the section above.

The settlement function. To contrast the constructions against the underlying ledgers, it is convenient to introduce a settlement function $s(r)$, which provides an error bound for the event that a transaction submitted at a time t_0 has not (relatively) settled by time $t_0 + r$. Assuming that the underlying ledgers provide exponential liveness and persistence yields settlement

$$s(r) \leq \mathrm{p}_A(r/2) + \mathrm{l}(r/2) = \exp(-\Theta(r)) \quad \text{(settlement of underlying ledgers } \mathbf{D}_i\text{)}.$$

Our goal is to demonstrate that the fast ledger $\mathsf{P}_F[\mathcal{D}]$ provides linear amplification, yielding settlement function $\overline{s}$ of the form

$$\overline{s}_{\mathsf{P}_F}(r) \leq \exp(-\Theta(mr)) + \exp(-\tilde{\Theta}(r + L)) \quad \text{(settlement of fast ledger } \mathsf{P}_F[\mathcal{D}]\text{)}.$$

(Here the $\tilde{\Theta}()$ notation neglects an additive term linear in m but logarithmic in L and r.) Note that this scales as $\exp(-\Theta(rm))$ so long as $rm \leq L$.

As discussed earlier, participants are free to use the "slow" logging mechanism (that is, simply logging their transaction in a single of the underlying ledgers), in which case they will achieve

$$\overline{s}_{\mathsf{P}}(r) \leq \exp(-\Theta(r) + O(L \ln m)) \qquad \text{(settlement of slow ledger } \mathsf{P}[\mathcal{D}]\text{)}.$$

Thus parameter L determines the transition between fast and slow settlement. For $r \approx L/m$, one achieves fast settlement; for $r \approx L \log m$, the system provides settlement guarantees asymptotically consistent with those of the underlying ledgers themselves.

Theorem 4 (Restatement of Theorem 2 for $\mathsf{P}_F[\mathcal{D}]$). *Let $\mathcal{D} = (\mathbf{D}_1, \ldots, \mathbf{D}_m)$ be a family of m subindependent dynamic ledgers defined over a common transaction space $\mathcal{T}$ with a conflict relation, each possessing exponential liveness $\mathrm{l}(r) = \exp(-\alpha_{\mathrm{l}} r + \beta_{\mathrm{l}})$ and absolute persistence $\mathrm{p}(r) = \exp(-\alpha_{\mathrm{p}} r + \beta_{\mathrm{p}})$. Consider the combined dynamic ledger $\mathsf{P}_F[\mathcal{D}]$ with the (preemptive) rank function $\overline{\mathsf{rank}}^*_L$ for a parameter $L \geq m$. We have*

$$\Pr\begin{bmatrix}\exists \mathrm{tx} \in A^{(t_0)} \text{ not relativ-}\\ \text{ely settled at time } t_0{+}2r\end{bmatrix} \leq \begin{array}{l}\exp(-r\Omega(m) + O(m))+\\ \exp\left(-\Omega(r) - \Omega(L\ln(m)) + O(m\ln(L+r))\right),\end{array}$$

thus there is a constant $C > 1$ so that if $L \geq Cm \ln m$ this probability is

$$\exp(-r\Omega(m) + O(m)) + \exp\left(-\Omega(r) - \Omega(L\ln(m))\right) .$$

The constants hidden in the $\Omega()$ and $O()$ notation depend on $\alpha_{\mathrm{p}}, \alpha_{\mathrm{l}}, \beta_{\mathrm{p}}, \beta_{\mathrm{l}}$ (and constants selected during the proof), but they are independent of m, L, and r.

Proof. Consider the set of transactions $A^{(t_0)}$. In light of Corollary 6, at time $T = t_0 + 2r$ these transactions will appear in at least $(1 - \gamma)m$ of the ledgers with rank $t_0 + r$ except with probability

$$\binom{m}{\gamma m} \exp(-\alpha_{\mathrm{l}} r + \beta_{\mathrm{l}})^{m\gamma} \leq \exp(-\gamma m[\alpha_{\mathrm{l}} r - \beta_{\mathrm{l}} - \ln(e/\gamma)]) .$$

Specifically the $(t_0 + r, r_0 + 2r)$-census of these transactions is at least $(1-\gamma)m$. Observe that so long as r exceeds a constant determined by α, β, and γ, this has the desired scaling.

We now consider the possibility that a transaction from $A^{(t_0+2r)}$ (or later) that conflicts with some transaction in $A^{(t_0)}$ can achieve rank less than those in $A^{(t_0)}$. We observe that almost all of the $(1-\gamma)m$ ledgers guaranteed above (that contain the transactions of $A^{(t_0)}$ at rank at most $t_0 + r$) are fixed for all future times up to this rank. Specifically, the probability that more than γm of these ledgers are not persistent through rank $t_0 + r$ (in the view of future times $T \geq t_0 + 2r$) is at most

$$\binom{m}{\gamma m} \exp(-\alpha_{\mathrm{p}} r + \beta_{\mathrm{p}})^{\gamma m} \leq \exp(-\gamma m[\alpha_{\mathrm{p}} r - \beta_{\mathrm{p}} - \ln(e/\gamma)])\,.$$

As above, for a constant r that depends only on α_{p}, β_{p}, and γ, we achieve the desired scaling.

Observe that—except with this small error probability $\exp(-\Omega(mr))$—all transactions in $A^{(t_0)}$ have $\overline{\mathsf{rank}}^*_L$ no more than $t_0 + r + \ln(1/(1-2\gamma))L$ at all future times.

In order for a transaction appearing after t_0+2r to compete with a transaction in $A^{(t_0)}$, then, it must achieve a $\overline{\mathsf{rank}}^*_L$ of $t_0 + r + \ln(1/(1-2\gamma))L$ using only $2\gamma m$ of the ledgers. At time $T = t_0 + 2r$, we apply Corollary 4 with the setting of $D = r - \ln(1/(1-2\gamma))L$; further assuming that $\gamma < 1/4$, this event can occur with probability no more than

$$(r + L\ln m)^{2\gamma m} \cdot \exp\left(-\alpha_{\mathrm{p}}\left[r - L\ln\left(\frac{1}{1-2\gamma}\right) + L\ln\left(m - \frac{2\gamma m}{1-2\gamma}\right)\right] + (\alpha_{\mathrm{p}} + \beta_{\mathrm{p}})2\gamma m\right)\,.$$

As we assume $m \leq L$, this is no more than

$$\begin{aligned}(r + L^2)^{2\gamma m} \exp\left(-\alpha_{\mathrm{p}} r - \alpha_{\mathrm{p}} L\left[\ln\left(m\frac{1-4\gamma}{1-2\gamma}\right) - 1\right] + (\alpha_{\mathrm{p}} + \beta_{\mathrm{p}})2\gamma m\right)\\ = \exp\left(-\alpha_{\mathrm{p}} r - \alpha_{\mathrm{p}} L\left[\ln(m) + O(1)\right] + O(m\ln(L+r))\right)\,.\end{aligned}$$

By choosing $L = Cm\log m$ for large enough C, we obtain the form recorded in the statement of the theorem. □

Remark 1. By setting $\gamma = 1/5$ in the proof above, we obtain a version that reflects the leading constants in the exponent. The three contributing terms are:

$\exp(-(m/5)[\alpha_{\mathrm{l}} r - (\beta_{\mathrm{l}} + 3)])$	Failure of $A^{(t_0)}$ to achieve (t_0+r, t_0+2r)-census $\geq 4m/5$;
$\exp(-(m/5)[\alpha_{\mathrm{p}} r - (\beta_{\mathrm{p}} + 3)])$	Persistence failure exceeding $m/5$ of these transactions at rank $t_0 + r$;
$\exp(-\alpha_{\mathrm{p}} r - \alpha_{\mathrm{p}} L[\ln(\frac{m}{3e})] + \frac{m}{5}(2\beta_{\mathrm{p}} + 4\ln(r+L))$	Persistence failure of remaining rank by insertion into $2m/5$ chains.

3.7.1 Worst-Case Constant-Time Settlement

In the setting where we have the luxury to select m so that it scales with the security parameter of the system, the construction above provides *constant time* settlement. Specifically, examining the statement (and following remarks with explicit bounds) of Theorem 2 above, by merely taking r large enough to ensure that $\alpha_{\mathrm{l}} r \geq \beta_{\mathrm{l}} + 4$ and $\alpha_{\mathrm{p}} r \geq \beta_{\mathrm{p}} + 4$ the first two failure terms above both decay exponentially in m. Likewise, by suitably adjusting L so that

$$L \geq \frac{m + (m/5)(2\beta_{\mathrm{p}} + 4\ln(r+L))}{\alpha_{\mathrm{p}} \ln(m/3e)}$$

the third term also falls off exponentially in m. (This is always possible with $L = O(m \log m)$.) Thus this achieves settlement in constant time except with probability negligible in the security parameter, establishing the following corollary stated earlier.

Corollary 1 (restated). *In the setting of Theorem 2, if the number of chains m scales with the security parameter then $\mathsf{P}_F[\mathcal{D}]$ achieves constant-time settlement except with an error probability negligible in the security parameter.*

In the full paper [17] we additionally explore the amplification problem in a stronger setting, called the *coordinated model*, assuming that any transaction attempted to be included into any of the ledgers is also immediately attempted to be included into all of the remaining ledgers. This allows to adopt a simpler rank function and achieve simpler results.

4 Implementation Considerations

4.1 Achieving Subindependence

Proof of Stake. Subindependence is easier to achieve in the proof-of-stake setting. In PoS, block creation rights are attributed to protocol participants via a stake-based lottery governed by randomness that is derived as a part of the protocol. Hence, a straightforward solution for obtaining (sub)independence in a setup with m PoS blockchains is to derive independent lottery randomness for selecting block creators for each of the chains (even in situations where these are sampled from the same stake distribution). This approach has been proposed before, e.g., in [19], and hence we omit the details.

Proof of Work. Blockchain subindependence in the proof-of-work setting can be achieved by generalizing the 2-for-1-PoW idea from [20] where two independent PoW-oracle queries are obtained from a single invocation of the random oracle. Similarly to [5], we propose a construction for an m-for-1-PoW to achieve m PoW-queries (one for each chain) by invocation of one single random oracle query—however, introducing some dependence between the m resulting queries. Still, the construction is sufficient to serve as a common PoW to maintain m *subindependent* ledgers.

The Construction. Given a hash function $H : \{0,1\}^* \to \{0,1\}^\kappa$ modeled as a random oracle, we partition a hash output $Y = H(X)$ into two bit-segments $Y = (Y_1, Y_2)$ of size $\kappa/2$ each. The first segment decides whether the query is *successful* (by the test $Y_1 < T$ for some threshold T with $p \stackrel{\text{def}}{=} T/2^{\kappa/2}$), the second segment *assigns* the invocation to a particular PoW instance $i \in [m]$ (by computing $i = 1 + (Y_2 \bmod m)$). The single invocation $H(X)$ is then defined to be *successful for instance* i if it is both successful and is assigned to instance i (i.e., $Y_1 < T$ and $i = 1 + (Y_2 \mod m)$). Formally, we write $\mathsf{PoW}_p^m(X) \stackrel{\text{def}}{=} (S_1, \ldots, S_m)$ where $S_i \stackrel{\text{def}}{=} (Y_1 < T \wedge i = 1 + (Y_2 \bmod m)) \in \{0,1\}$ for the bit vector of successes of the query X with respect to all instances. Note that the random variables S_i are fully determined by X and the internal randomness of the random oracle.

Analysis. We compare $\mathsf{PoW}_p^m(X)$ to an "ideal" oracle $\mathsf{IPoW}_{p'}^m(X)$ that for each new query X samples a fresh response $\mathsf{IPoW}_{p'}^m(X) \stackrel{\text{def}}{=} (\tilde{S}_1, \ldots, \tilde{S}_m)$ such that each binary random variable $\tilde{S}_i$ takes value 1 with probability p' and all $\tilde{S}_i$ are independent; repeated queries are answered consistently. Responses to new queries $\mathsf{IPoW}_{p'}^m(X)$ hence also depend only on the input and the internal randomness of $\mathsf{IPoW}_{p'}^m$. Let $\delta(\cdot,\cdot)$ denote the standard notion of statistical distance (sometimes called the total variation distance) of random variables. Then we have the following simple observation.

Lemma 7. *For any $x \in \{0,1\}^*$ and $p \in (0,1)$, we have*

$$\delta\left(\mathsf{PoW}_p^m(x), \mathsf{IPoW}_{p/m}^m(x)\right) \leq p^2 \ .$$

The above lemma already justifies the use of PoW_p^m for achieving subindependence in practical scenarios. To observe this, note that the use of $\mathsf{IPoW}_{p/m}^m$ would lead to full independence of the individual PoW lotteries, and by Lemma 7 the real execution with PoW_p^m will only differ from this ideal behavior with probability at most $Q \cdot p^2$, where Q is the total number of PoW-queries. With current values of $p \approx 10^{-22}$ in, e.g., Bitcoin,[6] and the block creation time adjusting to 10 minutes, this difference would manifest on expectation in about 10^{18} years. Note that any future increase of the total mining difficulty while maintaining the block creation time would only increase this period. Nonetheless, in the full paper [17], we prove the following, fully-parameterized result.

Lemma 8. *Consider the collection of m dynamic ledgers $\mathcal{D} = (\mathbf{D}_1, \ldots, \mathbf{D}_m)$ produced by a parallel m-fold execution of Bitcoin using PoW_p^m as the joint PoW oracle as described above, with n parties, each making q queries to PoW_p^m per round. Let λ denote a security parameter and assume throughout that $q \geq \lambda^5$, $m \leq \lambda$, $pq \leq \lambda$, and the honest parties dominate the adversarial parties sufficiently to invoke existing analysis yielding exponential persistence and liveness bounds for an individual chain. Then the ledgers $\mathcal{D}$ satisfy ε-subindependence with $\varepsilon = \mathsf{poly}(\lambda) \cdot \exp(-\Omega(\lambda))$.*

[6] `https://btc.com/stats/diff`

4.2 Realizing Rank via Timestamped Blockchains

An important consideration when deploying our virtual ledger construction over existing blockchains is how to realize the notion of rank. We note that typical Nakamoto-style PoS blockchains (e.g., the Ouroboros family, Snow White) assume a common notion of time among the participants and explicitly label blocks with slot numbers with a direct correspondence to absolute time. These slot numbers (or, preferably, a notion of common time associated with each slot number) directly afford a notion of rank that provides the desired persistence and liveness guarantees. To formalize this property, we introduce the notion of a timestamped blockchain.

Definition 11. *A* timestamped blockchain *is one satisfying the following conventions:*

- Block timestamps. *Every block contains a declared timestamp.*
- Monotonicity. *In order for a block to be considered valid, its timestamp can be no less than the timestamps of all prior blocks in the blockchain. (Thus valid blockchains consist of blocks in monotonically increasing order.)*

Informally, we say that an algorithm is a timestamped blockchain algorithm *if it calls for participants to broadcast timestamped blockchains and to "respect timestamps." More specifically, the algorithm satisfies the following:*

- Faithful honest timestamping. *Honest participants always post blocks with timestamps determined by their local clocks.*
- Ignore future blocks. *Honest participants ignore blocks that contain a timestamp which is greater than their local time by more than a fixed constant. (These blocks might be considered later when the local clock of the participant "catches up" with the timestamp.)*

As mentioned above, typical Nakamoto-style PoS blockchains are timestamped by design. For PoW blockchains the situation varies case by case. For instance, Bitcoin provides block timestamps, but these follow a more complex convention which guarantees that the timestamp associated with each block exceeds the *median* timestamp of the previous 11 blocks. Note, then, that one can assign a "logical timestamp" to block B_t equal to the maximum timestamp on the blocks $\{B_i : i \leq t\}$; these logical timestamps are then monotonically non-decreasing. Ignoring future blocks is also a part of the Bitcoin protocol.

For blockchains that do not provide timestamps satisfying the above notion natively, the full paper [17] describes a straightforward transformation that modifies any longest-chain rule blockchain algorithm into a timestamped blockchain, and demonstrates security of the transformation.

Timestamped blockchains as dynamic ledgers. Timestamped blockchains can be interpreted as dynamic ledgers in the natural way: for a fixed party P and time t, the ledger $\mathbf{L}^{(t)}$—corresponding to the index t in the dynamic ledger—consists of all the transactions present in the blocks constituting the

blockchain $B_{\mathsf{P},t}$ held by P at time t that have a timestamp not greater than t. The rank of each transaction $\mathsf{tx} \in \mathbf{L}^{(t)}$ is then defined to be the timestamp of the earliest block in $B_{\mathsf{P},t}$ containing it. Observe that standard exponentially vanishing error bounds on the persistence and liveness of such blockchains then translate to exponential failure bounds for the respective properties of the dynamic ledger.

4.3 A Proof-of-Work Instantiation

In this section we summarize the implications of our results for the proof-of-work setting by proving Corollary 7, which is a more detailed version of Theorem 1. Recall the definition of the (p, q)-flat PoW model from Theorem 1.

Corollary 7. *Let $\epsilon > 0$ and let λ denote the security parameter. Let $\mathcal{D} = (\mathbf{D}_1, \ldots, \mathbf{D}_m)$ be a family of $m = \lambda$ dynamic ledgers induced from m PoW-based blockchains using PoW_p^m as their joint PoW oracle, having a common transaction space with a conflict relation, and run by a combined population of $n = \mathsf{poly}(\lambda)$ parties in the synchronous (p, q)-flat PoW model, out of which at least a $(1/2+\epsilon)$-fraction is honest. Let the assumptions of Lemma 8 be satisfied, i.e., $q \geq \lambda^5$, $pq \leq \lambda$.*

Consider the combined dynamic ledger $\mathsf{P}_F[\mathcal{D}]$ with the (preemptive) rank function $\overline{\mathsf{rank}}_L^*$. Then for $\mathsf{P}_F[\mathcal{D}]$, there is a constant $C > 1$ so that if $L = Cm \ln m$, $\mathsf{P}_F[\mathcal{D}]$ achieves constant-time relative settlement except with an error probability negligible in the security parameter λ. (Observe that with such choice of L the system still provides meaningful single-chain settlement guarantees.)

Proof (sketch). The statement is an instantiation of Corollary 1 (which is itself based on Thm. 2) to the case of $m = \lambda$ PoW-based ledgers using a joint PoW_p^m oracle. The required subindependence of this mining mechanism follows from Lemma 8. □

References

1. S. Arora, E. Hazan, and S. Kale. The multiplicative weights update method: a meta-algorithm and applications. *Theory of Computing*, 8(6):121–164, 2012.
2. C. Badertscher, P. Gaži, 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.
3. C. Badertscher, P. Gaži, A. Kiayias, A. Russell, and V. Zikas. Ouroboros chronos: Permissionless clock synchronization via proof-of-stake. Cryptology ePrint Archive, Report 2019/838, 2019. `https://eprint.iacr.org/2019/838`.
4. 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.
5. V. K. Bagaria, S. Kannan, D. Tse, G. C. Fanti, and P. Viswanath. Prism: Deconstructing the blockchain to approach physical limits. In L. Cavallaro, J. Kinder, X. Wang, and J. Katz, editors, *ACM CCS 2019*, pages 585–602. ACM Press, Nov. 2019.

6. D. Boneh and X. Boyen. On the impossibility of efficiently combining collision resistant hash functions. In C. Dwork, editor, *CRYPTO 2006*, volume 4117 of *LNCS*, pages 570–583. Springer, Heidelberg, Aug. 2006.
7. V. Buterin. A next-generation smart contract and decentralized application platform. 2009. Online manuscript.
8. R. Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *42nd FOCS*, pages 136–145. IEEE Computer Society Press, Oct. 2001.
9. J. Chen and S. Micali. Algorand, 2016. arXiv preprint 1607.01341.
10. B. Cohen and K. Pietrzak. The chia network blockchain, 2019. Online manuscript.
11. P. Daian, R. Pass, and E. Shi. Snow white: Robustly reconfigurable consensus and applications to provably secure proof of stake. In I. Goldberg and T. Moore, editors, *FC 2019*, volume 11598 of *LNCS*, pages 23–41. Springer, Heidelberg, Feb. 2019.
12. B. David, P. Gaži, 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.
13. Y. Dodis, R. Impagliazzo, R. Jaiswal, and V. Kabanets. Security amplification for interactive cryptographic primitives. In O. Reingold, editor, *TCC 2009*, volume 5444 of *LNCS*, pages 128–145. Springer, Heidelberg, Mar. 2009.
14. S. Dziembowski, L. Eckey, S. Faust, and D. Malinowski. Perun: Virtual payment hubs over cryptocurrencies. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 327–344, 2019.
15. S. Dziembowski, S. Faust, and K. Hostáková. General state channel networks. In D. Lie, M. Mannan, M. Backes, and X. Wang, editors, *ACM CCS 2018*, pages 949–966. ACM Press, Oct. 2018.
16. M. Fischlin and A. Lehmann. Security-amplifying combiners for collision-resistant hash functions. In A. Menezes, editor, *CRYPTO 2007*, volume 4622 of *LNCS*, pages 224–243. Springer, Heidelberg, Aug. 2007.
17. M. Fitzi, P. Gaži, A. Kiayias, and A. Russell. Ledger combiners for fast settlement, 2020. Cryptology ePrint Archive, Report 2020/675.
18. M. Fitzi, P. Gaži, A. Kiayias, and A. Russell. Parallel chains: Improving throughput and latency of blockchain protocols via parallel composition, 2018. Cryptology ePrint Archive, Report 2018/1119.
19. M. Fitzi, P. Gaži, A. Kiayias, and A. Russell. Proof-of-stake blockchain protocols with near-optimal throughput, 2020. Cryptology ePrint Archive, Report 2020/037.
20. 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.
21. P. Gaži and U. Maurer. Free-start distinguishing: Combining two types of indistinguishability amplification. In K. Kurosawa, editor, *ICITS 09*, volume 5973 of *LNCS*, pages 28–44. Springer, Heidelberg, Dec. 2010.
22. D. Harnik, J. Kilian, M. Naor, O. Reingold, and A. Rosen. On robust combiners for oblivious transfer and other primitives. In R. Cramer, editor, *EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 96–113. Springer, Heidelberg, May 2005.
23. A. Herzberg. On tolerant cryptographic constructions. In A. Menezes, editor, *CT-RSA 2005*, volume 3376 of *LNCS*, pages 172–190. Springer, Heidelberg, Feb. 2005.
24. 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.

25. S. Lerner. Dagcoin draft, 2015. Online manuscript.
26. M. Luby and C. Rackoff. Pseudo-random permutation generators and cryptographic composition. In *18th ACM STOC*, pages 356–363. ACM Press, May 1986.
27. B. Magri, C. Matt, J. B. Nielsen, and D. Tschudi. Afgjort: A partially synchronous finality layer for blockchains, 2019. Cryptology ePrint Archive, Report 2019/504.
28. U. M. Maurer, Y. A. Oswald, K. Pietrzak, and J. Sjödin. Luby-Rackoff ciphers from weak round functions? In S. Vaudenay, editor, *EUROCRYPT 2006*, volume 4004 of *LNCS*, pages 391–408. Springer, Heidelberg, May / June 2006.
29. U. M. Maurer and K. Pietrzak. Composition of random systems: When two weak make one strong. In M. Naor, editor, *TCC 2004*, volume 2951 of *LNCS*, pages 410–427. Springer, Heidelberg, Feb. 2004.
30. U. M. Maurer, K. Pietrzak, and R. Renner. Indistinguishability amplification. In A. Menezes, editor, *CRYPTO 2007*, volume 4622 of *LNCS*, pages 130–149. Springer, Heidelberg, Aug. 2007.
31. U. M. Maurer and S. Tessaro. Computational indistinguishability amplification: Tight product theorems for system composition. In S. Halevi, editor, *CRYPTO 2009*, volume 5677 of *LNCS*, pages 355–373. Springer, Heidelberg, Aug. 2009.
32. S. Myers. Efficient amplification of the security of weak pseudo-random function generators. *Journal of Cryptology*, 16(1):1–24, Jan. 2003.
33. S. Nakamoto. Bitcoin: A peer-to-peer electronic cash system, 2008. Online manuscript.
34. S. Park, A. Kwon, G. Fuchbauer, P. Gazi, J. Alwen, and K. Pietrzak. Spacemint: A cryptocurrency based on proofs of space. In *Proceedings of the 22nd International Conference on Financial Cryptography and Data Security (FC). Springer*, 2018.
35. R. Pass, L. Seeman, and a. shelat. Analysis of the blockchain protocol in asynchronous networks. In J. Coron and J. B. Nielsen, editors, *EUROCRYPT 2017, Part II*, volume 10211 of *LNCS*, pages 643–673. Springer, Heidelberg, Apr. / May 2017.
36. R. Pass and E. Shi. Hybrid consensus: Efficient consensus in the permissionless model, 2016. Cryptology ePrint Archive, Report 2016/917.
37. 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.
38. R. Pass and E. Shi. Thunderella: Blockchains with optimistic instant confirmation. In J. B. Nielsen and V. Rijmen, editors, *EUROCRYPT 2018, Part II*, volume 10821 of *LNCS*, pages 3–33. Springer, Heidelberg, Apr. / May 2018.
39. K. Pietrzak. Composition does not imply adaptive security. In V. Shoup, editor, *CRYPTO 2005*, volume 3621 of *LNCS*, pages 55–65. Springer, Heidelberg, Aug. 2005.
40. K. Pietrzak. Non-trivial black-box combiners for collision-resistant hash-functions don't exist. In M. Naor, editor, *EUROCRYPT 2007*, volume 4515 of *LNCS*, pages 23–33. Springer, Heidelberg, May 2007.
41. J. Poon and T. Dryja. The bitcoin lightning network: Scalable off-chain instant payments, 2016. Online manuscript.
42. Y. Sompolinsky, Y. Lewenberg, and A. Zohar. SPECTRE: A fast and scalable cryptocurrency protocol, 2016. Cryptology ePrint Archive, Report 2016/1159.
43. S. Vaudenay. Decorrelation: A theory for block cipher security. *Journal of Cryptology*, 16(4):249–286, Sept. 2003.