Tight Proofs of Space and Replication

Ben Fisch

Stanford University

Abstract. We construct a concretely practical *proof-of-space* (PoS) with arbitrarily *tight* security based on stacked depth robust graphs and constant-degree expander graphs. A *proof-of-space* (PoS) is an interactive proof system where a prover demonstrates that it is persistently using space to store information. A PoS is arbitrarily tight if the honest prover uses exactly N space and for any $\epsilon > 0$ the construction can be tuned such that no adversary can pass verification using less than $(1-\epsilon)N$ space. Most notably, the degree of the graphs in our construction are independent of ϵ, and the number of layers is only $O(\log(1/\epsilon))$. The proof size is $O(d/\epsilon)$. The degree d depends on the depth robust graphs, which are only required to maintain $\Omega(N)$ depth in subgraphs on 80% of the nodes. Our tight PoS is also secure against parallel attacks.
Tight proofs of space are necessary for *proof-of-replication* (PoRep), which is a publicly verifiable proof that the prover is dedicating unique resources to storing one or more retrievable replicas of a specified file. Our main PoS construction can be used as a PoRep, but data extraction is as inefficient as replica generation. We present a second variant of our construction called ZigZag PoRep that has fast/parallelizable data extraction compared to replica generation and maintains the same space tightness while only increasing the number of levels by roughly a factor two.

1 Introduction

Proof-of-space (PoS) has been proposed as an alternative to proof-of-work (PoW) for applications such as SPAM prevention, DOS attacks, and Sybil resistance in blockchain-based consensus mechanisms [8, 11, 16]. Several industry projects[1] are underway to deploy cryptocurrencies similar to Bitcoin that use proof-of-space instead of proof-of-work. Proof-of-space is promoted as more egalitarian and eco-friendly that proof-of-work because it is ASIC-resistant and does not consume its resource (space instead of energy), but rather reuses it.

A PoS is an interactive protocol between a prover and verifier in which the prover uses a minimum specified amount of space in order to pass verification. The protocol must have compact communication relative to the

[1] `https://chia.net/`,`https://spacemesh.io/`, `https://filecoin.io/`

prover's space requirements and efficient verification. A PoS is *persistent* if repeated audits force the prover to utilize this space over a period of time. More precisely, there is an "offline" phase in which the prover obtains challenges from a verifier, generates a (long) string σ that it stores, and outputs a compact verification tag τ to the verifier. (The offline phase can be made non-interactive using the Fiat-Shamir transform). This is followed by an "online" challenge-response protocol in which the verifier uses τ to generate challenges and the prover uses σ to efficiently compute responses to the verifier's challenges.

The soundness of the PoS relies on a time bound on the online prover that is enforced by frequent verifier audits. A time bound is necessary as otherwise the prover could store its compact transcript and simulate the setup to re-derive the advice whenever it needs to pass an online proof. If the PoS guarantees that an adversary must use the minimum amount of space to pass challenges within the wall-clock time allotted no matter how much (polynomially bounded) computation it expends then the PoS is said to resist parallelization attacks. A PoS must resist parallelization attacks in order to be considered unconditionally secure. Otherwise, the security may still be reasoned through a cost benefit analysis for a *rational* prover, who will not expend significant computation to save a relatively small fraction of space.

More formally, correctness and (S, T, μ)-soundness for a PoS protocol is defined as follows. First, if the prover commits to persistently utilize N blocks of space then the honest prover algorithm defined by the protocol must use $O(N)$ persistent space and must succeed in passing the verifier's challenges without error. Next, a pair of offline/online adversaries is considered. The "offline" adversary generates an adversarial string σ' and offline proof π'. An (S, T, μ)-sound protocol guarantees that if the string length of σ' output by the online adversary is less than S then either the verifier accepts π' with negligible probability or otherwise any online adversary who runs in time less than T on the input σ' and the verifier's challenge will fail verification with probability at least $1 - \mu$.

There is generally a gap between the honest space utilization and the lower bound S on the adversary's space. If the honest prover uses S' space and some adversary is able to use $(1 - \epsilon)S'$ space then this PoS protocol has at least an ϵ *space gap*. Loosely speaking, a *tight PoS* construction makes ϵ arbitrarily small. The construction is allowed to involve ϵ as a parameter, and the value of ϵ may impact efficiency. All else equal, a tighter PoS is obviously more desirable as it has tighter provable security. Nearly all existing PoS constructions have enormous space gaps, including those that are currently being used in practice [2, 8]. The one exception is a recent PoS protocol by Pietrzak [18]. Although this PoS construction is provably tight, the concrete parameters required in the analysis result in an impractically large offline proof.

Proof-of-replication (PoRep) [1, 9, 10, 18] is a recently proposed variant of PoS. A PoRep demonstrates that the prover is dedicating unique resources to storing a retrievable copy of a committed data file, and is therefore a *useful proof of space*. It has been proposed as an alternative Sybil resistance mechanism (e.g. for a blockchain) that is not only ASIC resistant and eco-friendly, but also has a useful side-effect: file storage. Furthermore, since the prover may run several independent PoReps for the same file that each require unique resources, PoReps may be used as a publicly verifiable proof of data replication/duplication.

Unfortunately, it is not possible to cryptographically guarantee that a prover is persistently storing data in a replicated format. A prover can always sabotage the format (e.g. by encrypting it and storing the key separately) and can then recover the original format quickly when challenged. A recently proposed security model for PoReps is *ϵ-rational replication* [9], which says that an adversary can save at most an ϵ fraction of its space by deviating from storing the data in a replicated format. A PoRep that satisfies ϵ-rational replication is also a PoS with an ϵ space gap. Intuitively, if a PoRep is not a tight proof of space then there may exist some adversary that would be rationally incentivized to deviate from honest behavior in a way that also destroys the replication format. In fact, if the input file is incompressible then any adversary who manages to saves an ϵ fraction of the claimed space cannot be storing the data in the replicated format. Thus, tight proofs of space are necessary for PoReps because a PoRep construction is only meaningfully secure when ϵ is very small.

The goal of this work is to construct a practical and provably tight PoS that can also be used as a PoRep that satisfies ϵ-rational replication for arbitrarily small ϵ.

1.1 Related work

The original PoS of Dziembowski et. al. [8] was based on hard to pebble directed acyclic graphs (DAGs), using a blend of techniques from superconcentrators, random bipartite expander graphs and depth robust graphs [17]. During the offline initialization the prover computes a *labeling* of the graph using a collision-resistant hash function where the label e_v on each node $v \in \mathcal{G}$ of the graph is the output of the hash function on the labels of all parent nodes of v. It outputs a commitment to this labeling along with a proof that the committed labeling was "mostly" correct. This offline proof consists of randomly sampled labels and their parent labels, which the verifier checks for consistency. During the online challenge-response phase the verifier simply asks for random labels that the prover must produce along with a standard proof that these labels are consistent with the commitment. The construction leaves a space gap of at least $1 - \frac{1}{512}$.

Ren and Devadas [19] construct a PoS from stacked bipartite expander graphs that dramatically improved on the space gap, although it is not secure against parallel attacks. Their construction involves λ levels $V_1, ..., V_\lambda$ consisting of n nodes each, with edges between the layers defined by the edges of a constant-degree bipartite expander. The prover computes a labeling of the graph just as in the Dziembowski et. al. PoS, however it only stores the labels on the final level. Their construction still leaves a space gap of at least[2] 1/2.

Recently, Abusalah et. al. [2] revived the simple PoS approach based on storing tables of random functions. The basic idea is for the prover to compute and store the function table of a random function $f : [N] \rightarrow [N]$ where f is chosen by the verifier or a random public challenge. During the online challenge-response the verifier asks the prover to invert f on a randomly sampled point $x \in [n]$. Intuitively, a prover who has not stored most of the function table will likely have to brute force $f^{-1}(x)$, performing $\Omega(N)$ work. This simple approach fails to be a PoS due to Hellman's time/space tradeoffs, which enable a prover to succeed with S space and T computation for any $ST = O(N)$. However, Abusalah et. al. build on this approach to achieve a provable time/space tradeoff of $S^k T = \Omega(\epsilon^k N^k)$. This PoS is not secure against parallel attacks, and also has a very large (even asymptotic) space gap.

Pietrzak [18] and Fisch et. al [9, 10] independently proposed simpler variants of the graph labeling PoS by Dziembowski et. al. based solely on pebbling a depth robust graph (DRG). A degree d DAG on n nodes is (α, β)-depth robust if any subgraph on αn nodes contains a path of at least length βn. It is trivial to construct DRGs of large degree (a complete DAG is depth robust), but much harder to construct DRGs with small degree. Achieving constant α, β is only possible asymptotically with degree $\Omega(\log N)$. The graph labeling PoS on a DRG results in a PoS with

[2] For practical parameters, the Ren and Devadas construction has a space gap larger than 1/2. For example, it requires a graph of degree at least 40 in order to achieve a space gap of less than 2/3.

an α space gap that is also secure against parallel attacks. Fisch et. al. also suggested combining this labeling PoS with a *verifiable delay function* (VDF) [5] to increase the expense of labeling the graph without increasing the size of the proof verification complexity. The delay on the VDF can be tuned depending on the value of n. Both of these constructions were proposed for PoReps. In this variant of the PoS protocol, the prover uses the labeling of the graph to encode a data file on n blocks $D = d_1, ..., d_n$. The ith label e_i is computed by first deriving a key k_i by hashing the labels on the parents of the ith node, and then setting $e_i = k_i \oplus d_i$. If all the labels are stored then any data block can be quickly extracted from e_i by recomputing k_i. More generally, this *DAG encoding* of the data input could use any encoding scheme $(\mathsf{enc}, \mathsf{dec})$, where enc is sequentially slow and dec is fast, in order to derive $e_i = \mathsf{enc}(k_i, d_i)$. The data is decoded by computing $d_i = \mathsf{dec}(k_i, e_i)$.

The labeling PoS on a DRG is not technically a tight PoS because decreasing α also decreases the time bound βn on the prover's required computation to defeat the PoS. Moreover, while there exist constructions of (α, β)-DRGs for arbitrarily small α, these constructions have concretely very high degrees and are thus not useful for building a practical PoS. Pietrzak [18] improved on the basic construction by relying on a stronger property of special DRGs [4, 17] that have degree $\Omega((\log n)/\epsilon)$ and are (α, β)-depth robust for all (α, β) such that $1 - \alpha + \beta \geq 1 - \epsilon$. This DRG can be constructed for any value of $\epsilon < 1$. In Pietrzak's PoS, the prover builds a DRG on $4n$ nodes and only stores the labels on the topologically last n nodes. This can similarly be used as a PoRep where the data is encoded only on the last level and the labels on previous levels are just used as keys. This PoRep has a slow data extraction time because extracting the data requires recomputing most of the keys from scratch, which is as expensive as the PoS initialization.

Pietrzak shows that a prover who deletes an ϵ' fraction of the labels on the last n nodes will not be able to re-derive them in fewer than n sequential steps. The value ϵ' can be made arbitrarily small, but at the expense of increasing the degree of the graph proportionally to $1/\epsilon'$. The resulting proof has asymptotic size $\Omega((\log N)/\epsilon^2)$. Moreover, although these special DRGs achieve asymptotic efficiency, their current analysis requires the graphs to have impractically large degrees. According to the analysis in [4], achieving just a $1/2$ space gap would require instantiating these graphs with degree at least $2,760 \log N$. The proof size is proportional to the graph degree, so to achieve the space gap $\epsilon = 1/2$ with soundness $\mu = 2^{-10}$ and $N = 2^{30}$ the proof size would be at least 26 MB.

Boneh et. al. [5] describe a simple PoRep (also a PoS) just based on storing the output of a *verifiable delay function* (VDF) on N randomly sampled points, which generalizes an earlier proposal by Sergio Demian Lerner [13]. This is in fact an arbitrarily tight PoS with very practical proof sizes (essentially optimal). However, the time complexity of initializing the prover's $O(N)$ storage is $O(N^2)$, and therefore is not practically feasible for large N. This construction is similar to the PoS based on storing function tables [2], but uses the VDF as a moderately hard (non-parallelizable) function on a much larger domain (exponential in the security parameter) and stores a random subset of its function table. The reason for the large initialization complexity is that the prover cannot amortize its cost of evaluating the VDF on the entire subset of points.

1.2 Summary of Contributions

We construct a new tight PoS based on graph labeling with asymptotic proof size $O(\log N/\epsilon)$ where ϵ is the achieved space gap. We can instantiate this construction with relatively weak[3] depth robust graphs that do not require any special properties other than retaining $\Omega(N)$ depth in subgraphs on some constant fraction of the nodes bounded away from 1 (e.g. our concrete analysis assumes 80%).

PoS from Stacked DRGs Our basic approach is a combination of the stacked bipartite expanders of Ren and Devadas [19] with depth robust graphs. Instead of stacking λ path graphs we stack $O(\log(1/\epsilon))$ levels of fixed-degree DRGs where ϵ is a construction parameter. We refer to this graph construction as Stacked DRGs. We are able to show that this results in a PoS that has only an ϵ space gap. Intuitively, the expander edges between layers amplify the dependence of nodes on the last layer and nodes on earlier layers so that deletion of a small ϵ fraction of node labels on the last level will require re-derivation of nearly all the node labels on the first several layers. Thus, since every layer is a DRG, recomputing the missing ϵ fraction of labels requires $\Omega(N)$ sequential computation. It is easy to see that this would be the case if the prover were only storing $(1-\epsilon)n$ labels on the last level and none of the labels on earlier levels, however the analysis becomes much more difficult when the prover is allowed to store any arbitrary $(1-\epsilon)n$ labels. This analysis is the main technical

[3] There is experimental evidence that a simple DRG construction with concretely small constant degree (even degree 2) has this property on a graph of size $N = 2^{20}$ [3].

contribution of this work. Concretely, we analyze the construction with an $(n, 0.80n, \Omega(n))$ DRG, i.e. deletion of 20% of nodes leaves a high depth graph on the 80% remaining nodes, regardless of the value of ϵ.

Our construction is efficient compared to prior constructions of tight PoS primarily because we can keep the degree of the graphs fixed for arbitrary ϵ while keeping the number of levels proportional to $\log(1/\epsilon)$. In a graph labeling PoS, the offline PoS proofs sample $O(1/\epsilon)$ labels along with their parent labels, which the verifier checks for consistency. Thus, any construction based on this approach that requires scaling the degree of graphs by $1/\epsilon$ also scales the proof size by $1/\epsilon$, resulting in a proof complexity of at least $O(1/\epsilon^2)$. In our stacked DRG PoS construction the offline proof must include queries from each level to prove that each level of computed labels are "mostly" correct. If done naively, $O(1/\epsilon)$ challenge labels are sampled from each level, resulting in a proof complexity $O(d/\epsilon \cdot \log(1/\epsilon))$ where d is the degree of the level graphs. This is already an improvement, however with a more delicate analysis we are able to go even further and show that the total number of queries over all layers can be kept at $O(1/\epsilon)$, achieving an overall proof complexity $O(d/\epsilon)$.[4]

The PoS on Stacked DRGs can also be used as the basis for a PoRep that satisfies ϵ-rational replication for arbitrarily small ϵ. The PoRep simply uses the labels on the $\ell - 1$st level as keys to encode the n-block data input $D = d_1, ..., d_n$ on the ℓth (last) level, using the same method described earlier for encoding data into the labels of a PoS (see Related Work, [9, 10, 18]). However, extracting data from this PoRep is as expensive as initializing the PoRep space because it requires recomputing the keys on the $\ell - 1$st level.

PoRep from ZigZag Expander DRGs Our second contribution is a variant of the PoS on Stacked DRGs that compromises slightly on initialization efficiency and proof size (requires doubling the number of levels for the same security guarantee) but improves the efficiency of extracting data when this is used as a PoRep. Instead of adding bipartite expander edge dependencies between the layers, these edges are mapped into each layer itself. Specifically, an edge from the ith node of one layer to the jth node of the next is replaced with edges between the ith and jth nodes in each layer. The directionality of these mapped edges alternates between layers, forming a "zig-zag". The only edges retained between layers are

[4] Asymptotically, this is close to the optimal proof complexity achievable for any PoS based on graph labeling that has an ϵ space gap. If the prover claims to be storing n labels and the proof queries less than $1/\epsilon$ then a random deletion of an ϵ fraction of these labels evades detection with probability at least $(1-\epsilon)^{1/\epsilon} \approx 1/e$.

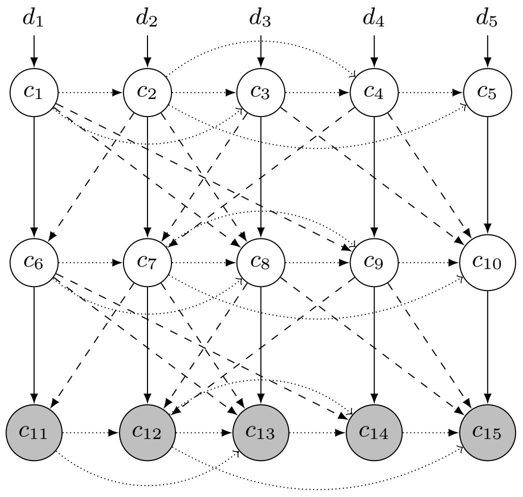

Fig. 1.1. Stacked DRGs. Dotted edges are the DRG edges and dashed edges are expander edges. In the PoS on Stacked DRGs the prover computes a labeling of the graph and stores the labels on the nodes in green.

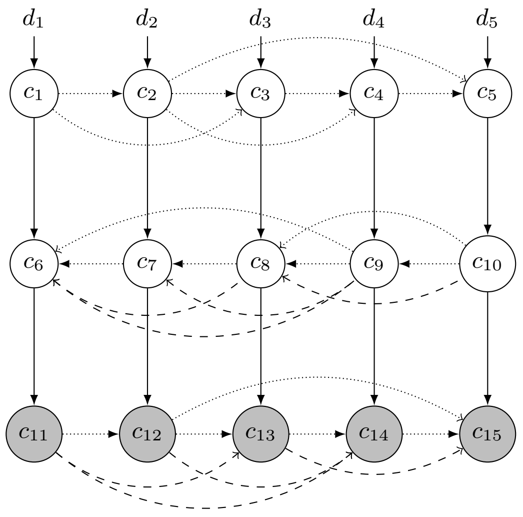

Fig. 1.2. ZigZag DRGs. The dashed edges in ZigZag DRGs are the same as in Stacked DRGs but projected into the layers. Dashed edges in ZigZag DRGs are reversed every other layer while dotted edges are redefined by reversing the order of the nodes. Dashed edges correspond to encoding instead of key-derivation dependencies. In the PoRep on ZigZag DRGs each labeling on a layer encodes the previous layer and the prover stores only the encoding labels of the green nodes.

between nodes at the same indices. As an undirected graph, every layer is the union of a DRG and a constant degree non-bipartite expander graph. As a directed graph, each layer forms a DAG where the union of any subset with its dependencies and targets is a constant fraction larger than the subset itself. By alternating the direction of the edges between layers, the dependencies of a subset in one layer become targets of the same subset in the adjacent layer, and the dependencies between layers expands. We refer to this graph construction as ZigZag DRGs.

The PoRep on ZigZag DRGs encodes in the labels of each layer the labels of the previous levels. The edges within a layer enforce dependencies between labels by deriving a key for each encoding using a cryptographic hash function. A special key is derived for the encoding on each ith node from the labels on the parents of the ith node within the same layer. Essentially, this construction on ℓ DAG layers iterates the basic DAG encoding of the data inputs ℓ times rather than performing a long key derivation. The labels in any given layer can be decoded (in parallel) from the labels in the preceding layer.

2 Preliminaries

2.1 Proofs of Space

A PoS interactive protocol has three procedures:

1. Setup The setup runs on security parameters λ and outputs public parameters pp for the scheme. The public parameters are implicit inputs to the next two protocols.
2. Initialization is an interactive protocol between a prover P and verifier V that run on shared input (id, N). P outputs Φ and S, where S is its storage advice of length N and Φ is a compact $O(\text{polylog}(N))$ length string given to the verifier.
3. Execution is an interactive protocol between P and V where P runs on input S and V runs on input Φ. V sends challenges to P, obtains back a proof π, and outputs accept or reject.

Efficiency. The commitment Φ is $O(\text{polylog}(N))$ size, the storage S is size N, and the verifier runs in time $O(\text{polylog}(N))$.

Completeness. The prover succeeds with probability 1 (causes verifier to accept) if it follows the protocol honestly.

Soundness. The PoS is (s, t, μ)-sound if for all adversaries P^* running in time t and storing advice of size s during $\mathsf{Execution}$, P^* passes verification with probability at most μ. The PoS is parallel (s, t, μ)-sound if P^* may run in parallel time t. We say that a PoS construction is *tight* if for any constant $\epsilon < 1$ the PoS can be parametrized so that the resulting PoS is $(\epsilon N, t, \mu)$-sound for $t \in \Omega(N)$ and $\mu = \mathsf{negl}(\lambda)$.

Amortization-free soundness An (s, t, μ) PoS is *amortization-free* if for any k distinct ids $id_1, ..., id_k$, the modified PoS protocol that runs $\mathsf{Initialization}$ on k independent inputs (id_i, N) for each i to get outputs (S_i, Φ_i) and then runs $\mathsf{Execution}$ independently on each (S_i, Φ_i) is $(ks, t, k\mu)$-sound.

2.2 Graph pebbling games

Pebbling games are the main analytical tool used in graph-based proofs of space and memory hard functions.

Black pebbling game The black pebbling game is a single-player game on a DAG $\mathcal{G} = (V, E)$. At the start of the game the player chooses a starting configuration of $P_0 \subseteq V$ of vertices that contain black pebbles. The game then proceeds in rounds where in each round the player may place a black pebble on a vertex only if all of its parent vertices currently contain pebbles placed in some prior round. In this case we say that the vertex is *available*. Placing a pebble constitutes a *move*, whereas placing pebbles on all simultaneously available vertices consumes a *round*. The adversary may also remove any black pebble at any point. The game stops once the adversary has placed pebbles on all vertices in some target/challenge set $V_C \subseteq V$.

Pebbling complexity The pebbling game on graph G with vertex set V and target set $V_C \subseteq V$ is (s, t)-*hard* if no player can pebble the set V_C in t moves (or fewer) starting from s initial pebbles, and is (s, t)-*parallel-hard* if no player can complete the pebbling in t rounds (or fewer) starting from an initial configuration of at most s pebbles. If a $1 - \alpha$ fraction of the nodes in V_C each require t rounds to pebble then the pebbling game on (G, V_C) is (s, t, α)-parallel-hard, i.e. every subset containing more than an α fraction of the nodes in VC requires t rounds to pebble.

In a random pebbling game a challenge node is sampled randomly from V_C after the player commits to the initial configuration P_0 of s vertices, and the hardness measure includes the adversary's probability of success. The random pebbling game is (s,t,ϵ)-(parallel)-hard if from any s fixed initial pebbles the probability that a uniformly sampled challenge node can be pebbled in t or fewer moves (resp. t or fewer rounds) is less than ϵ.

The following facts are easy to prove:

Fact 1 *The random pebbling game on a DAG G on n nodes with target set V_C is (s,t,α)-parallel-hard if and only if the deterministic pebbling game on G with target set V_C is (s,t,α)-parallel-hard.*

Fact 2 *A random pebbling game with a single challenge is (s,t,α)-parallel-hard if and only if the the random pebbling with κ challenges is (s,t,α^k)-parallel-hard.*

DAG labeling game A labeling game on a degree d DAG $\mathcal{G}$ is analogous to the pebbling game, but involves a cryptographic hash function $H:\{0,1\}^{dm}\rightarrow\{0,1\}^m$, often modeled as a random oracle. The vertices of the graph are indexed in $[n]$ and each ith vertex associated with the label c_i where $c_i = H(i)$ if i is a source vertex, or otherwise $c_i = H(i||c_{\mathsf{parents}}(i))$ where $c_{\mathsf{parents}}(i) = \{c_{v_1},...,c_{v_d}\}$ if $v_1,...,v_d$ are the parents of the ith vertex, i.e. the vertices with a directed edge to vertex i. The game ends when the player has computed all the labels on a target/challenge set of vertices V_C. A "fresh" labeling of $\mathcal{G}$ could be derived by choosing a salt id for the hash function so that $H_{id}(x) = H(id||x)$, and the labeling may be associated with the identifier id.

The complexity of the labeling game (on a fresh identifier id) is measured in queries to the hash function instead of pebbles. This includes the number of labels initially stored, the total number of queries, and the total rounds of sequential queries, etc. The labeling game is (s,r,q,ϵ,δ)-labeling-hard if no algorithm that stores initial advice of size s and after receiving a uniform random challenge node $v\in[n]$ makes a total of q queries to H in r sequential rounds can output the correct label on v with probability greater than ϵ over the challenge v and δ over the random oracle H.

Random oracle query complexity A general correspondence between the complexity of the black pebbling game on the underlying graph $\mathcal{G}$ and the random oracle labeling game is not yet known. However, Pietrzak [18] recently proved an equivalence between the parallel hardness of the randomized pebbling game and the parallel hardness of the random oracle labeling game for arbitrary initial configurations S_0 adapting the "ex post facto" technique from [7].

Theorem 1 (Pietrzak [18]). *If the random pebbling game on a DAG G with n nodes and in-degree d is (s, r, ϵ)-parallel-hard then the labeling game on G with a random oracle $H : \{0,1\}^{md} \to \{0,1\}^m$ is $(s', r, \epsilon, \delta, q)$-labeling-hard with $s' = s(m - 2(\log n + \log q)) - \log(1/\delta)$.*

Generic PoS from graph labeling game Many PoS constructions are based on the graph labeling game [8,18,19]. Let $\mathcal{G}(\cdot)$ be a family of d-in-regular DAGs such that $G_n \leftarrow \mathcal{G}(n)$ is a d-in-regular DAG on $N > n$ nodes and $V_C(n)$ is a subset of n nodes from G_n. Let $H : \{0,1\}^{dm} \to \{0,1\}^m$ be a collision-resistant hash function (or random oracle). Let $\mathsf{Chal}(n, \mathit{\Lambda})$ denote a distribution over challenge vectors in $[N]^\lambda$. For each $n \in \mathbb{N}$, the generic PoS based on the labeling game with G_n and target set $V_C(n)$ is as follows:

Initialization: The prover plays the labeling game on G_n using a hash function $H_{id} = H(id||\cdot)$. The prover does the following:

1. Computes the labels $c_1, ..., c_N$ on all nodes of $\mathcal{G}$ and commits to them in *com* using any vector commitment scheme.
2. Obtains vector of λ challenges $\boldsymbol{r} \overset{\mathrm{R}}{\leftarrow} \mathsf{Chal}(n)$ from the verifier (or non-interactively derives them using as a seed $H_{id}(com)$).
3. For challenges $r_1, ..., r_\lambda$, the prover opens the label on the r_ith node of G_n, which was committed in *com*, as well as the labels $c_{\mathsf{parents}}(r_i)$ of all its parent nodes. The labels are added to a list L with corresponding opening proofs in a list $\mathit{\Lambda}$ and the prover outputs the proof $\mathit{\Phi} = (com, L, \mathit{\Lambda})$.

The verifier checks the openings $\mathit{\Lambda}$ with respect to *com*. It also checks for each challenge specifying an index $v \in [N]$, the label c_v in L label and its parent labels $c_{\mathsf{parents}}(c_v)$, that $c_v = H_{id}(v||c_{\mathsf{parents}}(c_v))$. Finally, the prover stores as S only the n labels in V_C.

Execution: The verifier selects κ challenge nodes $v_1, ..., v_\kappa$ uniformly at random from V_C. The online prover uses its input S to respond with the label on v and an opening of *com* at the appropriate index. The verifier can repeat this sequentially, or ask for a randomly sampled vector of challenge vertices to amplify soundness.

Red-black pebbling game An adversary places both black and red pebbles on the graph initially. The red pebbles correspond to incorrect labels that the adversary computes during Initialization and the black pebbles correspond to labels the adversary stores in its advice S. Without loss of generality, an adversary that cheats generates some label that does not require any space to store, which is why red pebbles will be "free" pebbles and counted separately from black pebbles. The adversary's choice of red pebble placements (specifically how many to place in different regions of the graph) is constrained by the λ non-interactive challenges, which may catch these red pebbles and reveal them to the verifier. The formal description of the red-black pebbling security game for a graph labeling PoS construction with $\mathcal{G}(n)$, $V_C(n)$, and $\mathsf{Chal}(n)$ is as follows.

$\mathsf{Red\text{-}Black\text{-}Pebbles}^{\mathcal{A}}(\mathcal{G}, V_C, \mathsf{Chal}, t)$:

1. $\mathcal{A}$ outputs a set $R \subseteq [N]$ (of red pebble indices) and $S \subseteq [N]$ (of black pebble indices).
2. The challenger samples $c_1, ..., c_\lambda \overset{R}{\leftarrow} \mathsf{Chal}(n)$. If $c_i \in R$ for some i then $\mathcal{A}$ immediately loses. The challenger additionally samples $v_1, ..., v_\kappa$ uniformly at random from indices in $V_C(n)$ and sends these to $\mathcal{A}$.
3. $\mathcal{A}$ plays the random (black) pebbling game on $\mathcal{G}(n)$ with the challenges $v_1, ..., v_\kappa$ and initial pebble configuration $P_0 = R \cup S$. It runs for t parallel rounds and outputs its final pebble configuration P_t. $\mathcal{A}$ wins if P_t contains pebbles on all of $v_1, ..., v_\kappa$.

Graph labeling PoS soundness Given the correspondence between the hardness of the random oracle labeling game and parallel black pebbling game, we can entirely capture the soundness of the graph labeling PoS in terms of the complexity of $\mathsf{Red\text{-}Black\text{-}Pebbles}^{\mathcal{A}}(\mathcal{G}, V_C, t)$. Let $c : \mathbb{N} \to N$ denote a cost function $c : \mathbb{N} \to \mathbb{N}$ representing the parallel time cost (e.g. in sequential steps on a PRAM machine) of computing a label on a node of $\mathcal{G}(n)$ for each $n \in \mathbb{N}$.

Definition 1. *A graph labeling PoS with $\mathcal{G}(n), V_C(n), \mathsf{Chal}(n)$ and cost function $c(n)$ is parallel $(s, c(n) \cdot t, \mu)$-sound if and only if the probability that any $\mathcal{A}$ wins $\mathsf{Red\text{-}Black\text{-}Pebbles}^{\mathcal{A}}(\mathcal{G}, V_C, \mathsf{Chal}, t)$ is bounded by μ where $|S| = s$.*

2.3 Depth Robust Graphs

A directed acyclic graph (DAG) on n nodes with d-indegree is (n, α, β, d) *depth robust graph* (DRG) if every subgraph of αn nodes contains a path of length at least βn.

DRGs have been constructed for constant α, β and $d = O(\log n)$ using extreme constant-degree bipartite expander graphs, or local expanders [4, 14, 17]. Explicit constructions of local expanders exist [15], however they are complicated to implement and their concrete practicality is hindered by very large hidden constants. The most efficient way to instantiate these extreme expander graphs is probabilistically. A probabilistic DRG construction outputs a graph that is a DRG with overwhelming probability. The most efficient probabilistic construction to date is due to Alwen et. al. [3]. The analysis still leaves large gaps between security and efficiency although was shown to resist depth-reducing attacks empirically. Their construction is also *locally navigatable*, meaning that it comes with an efficient parent function to derive the parents of any node in the graph using polylogarithmic time and space.

2.4 Expander graphs

The vertex expansion of a graph $\mathcal{G}$ on vertex set V characterizes the size of the boundary of vertex subsets $S \subseteq V$ (i.e. the number of vertices in $V \setminus S$ that are neighbors with vertices in S). In the case of directed bipartite graphs, vertex expansion is defined by the minimum number of sources connected to any given number of sinks.

Definition 2. *For any constants α, β where $0 < \alpha < \beta < 1$ and integer $n \in \mathbb{N}$, an (n, α, β) bipartite expander is a directed bipartite graph with n sources and n sinks such that any subset of αn sinks are connected to at least βn sources. For any $\delta > 0$, a subset S of sinks is called $(1 + \delta)$-expanding if it is connected to at least $(1 + \delta)|S|$ sources.*

Chung's bipartite expander The randomized construction of Chung [6] defines the edges of a d-regular bipartite expander on $2n$ vertices by connecting the dn outgoing edges of the sources to the dn incoming edges of the sinks via a random permutation $\Pi : [d] \times [n] \to [d] \times [n]$. The ith source is connected to the jth sink if there is some $k_1, k_2 \in [d]$ such that $\Pi(k_1, i) = (k_2, j)$.

Lemma 1 (RD [19]). *The Chung random bipartite graph construction is a d-regular (n, α, β) expander with probability $1 - \mathsf{negl}(nH_b(\alpha))$ for all d, α, β satisfying:*

$$H_b(\alpha) + H_b(\beta) + d(\beta H_b(\alpha/\beta) - H_b(\alpha)) < 0 \tag{2.1}$$

where $H_b(x) = -x \log_2 x - (1 - x) \log_2(1 - x)$ is the binary entropy function.

For example, the above formula shows that for $\alpha = 1/2$ and $\beta = 0.80$ Chung's construction gives an $(n, 0.5, 0.80)$ expander for $d \geq 8$, meaning any subset of 50% of the sinks are connected to at least 80% of the sources when the degree is at least 8.

The following lemmas establish further properties of Chung's bipartite expander construction that will be used in the analysis of our PoS. The proofs are included in the full version of this paper. Let $\beta_G(\alpha)$ denote the smallest expansion of a subset of αn sources in a bipartite graph G, i.e. every subset of αn sources is connected to at least $\beta_G(\alpha)$ sinks.

Lemma 2. *For any $k > 1$ and $d > 2$, if the output of Chung's construction is a d-regular $(n, \alpha, k\alpha)$ bipartite expander for some $\alpha < \frac{d-k-1}{k(d-2)}$ with probability $1 - \mathsf{negl}(nH_b(\alpha))$ then $\beta_G(\alpha') \geq k\alpha'$ for every $\alpha' < \alpha$ with probability $1 - \mathsf{negl}(nH_b(\alpha'))$.*

Corollary 1. *For $d = 8$ Chung's construction is an 8-regular bipartite graph such that every subset of at most 1/3 of the nodes is 2-expanding, i.e. it is an $(n, \alpha, 2\alpha)$-bipartite expander for every $\alpha \leq 1/3$ with overwhelming probability.*

Proof. Plugging $\alpha = 1/3$ and $\beta = 2/3$ into the formula for degree (Equation 2.1) gives $d = 7.21 < 8$. With $d = 8$ and $k = 2$ the condition in Lemma 2 is satisfied: $\alpha = 1/3 < (d - k - 1)/k(d - 2) = 5/12$.

For fixed d the expansion improves further as α decreases. Figure 2.1 provides a table of expansion factors over a range of α with fixed degree $d = 8$. Figure 2.2 plots the expansion as a function of subset size.

Size (α)	0.01	0.10	0.20	0.30	0.40	0.45	0.50	0.55	0.60	0.65	0.70	0.75	0.80
Expansion (β)	0.04	0.33	0.53	0.65	0.75	0.78	0.81	0.84	0.88	0.89	0.91	0.93	0.94
Factor (β/α)	4	3.3	2.65	2.1	1.8	1.73	1.62	1.53	1.47	1.37	1.3	1.24	1.17

Fig. 2.1. A table of the maximum expansion (β) satisfying the condition from Lemma 1 for Chung's construction with fixed degree $d = 8$ over a range of subset sizes (α).

In a bipartite expanders, the "boundary" of a set of sources is the set of sinks connected to these sources that have distinct index labels from the sources, which is at least $\beta_G(\alpha) - \alpha$. Lemma 3 gives a smooth lower bound on $\beta_G(\alpha) - \alpha$ for Chung's bipartite expander graphs that we can show has a unique local maximum in $(0, 1)$.To simplify the analysis, we look at the function defined by the zeros of $\phi(\alpha, \beta) = d(\beta H_b(\alpha/\beta) - H_b(\alpha)) + 2 = 0$.

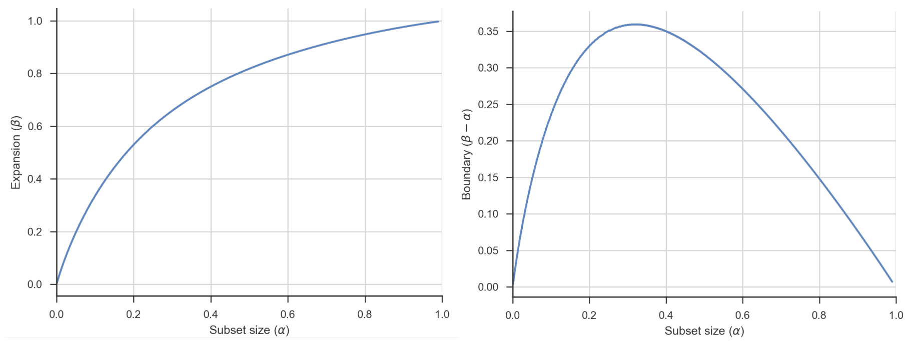

Fig. 2.2. The graph on the left plots the lower bound from Lemma 1 on the expansion β as a function of the subset size α (in fractions of the sources/sinks) for Chung's construction with fixed $d = 8$. The graph on the right plots the corresponding lower bound on $\beta - \alpha$, which is the analog of the subgraph boundary in non-bipartite expanders. Specifically, this is a lower bound on the fraction of sinks connected to an α fraction of sources that have distinct index labels from the sources.

Any α, β satisfying this relation also satisfies the relation in Lemma 1 because $H_b(\alpha) + H_b(\beta) < 2$ when $\beta > \alpha$. This implicitly defines β as a function of α, as well as the function $\hat{\beta} = \beta - \alpha$, by pairs of points $(\alpha, \hat{\beta}(\alpha))$ such that $\phi(\alpha, \alpha + \hat{\beta}(\alpha)) = 0$ is a lower bound to the boundary of subsets of size α, which holds at any point α with probability at least $1 - \mathsf{negl}(nH_b(\alpha))$.

Lemma 3. *Define $\phi(x, y) = d(yH_b(x/y) - H_b(x)) + c$ where c is any constant and let $\hat{\beta}$ be the function on $(0, 1)$ defined by pairs of points $(\alpha, \beta - \alpha)$ such that $\phi(\alpha, \beta) = 0$ and $0 < \alpha < \beta < 1$. The function $\hat{\beta}$ is continuously differentiable on $(0, 1)$ and has a unique local maximum.*

Corollary 2. *With overwhelming probability in n, Chung's construction (with $d = 8$) is an 8-regular bipartite graph on n sinks and n sources each indexed in $[n]$ such that for all $\alpha \in (0.10, 0.80)$ every αn sinks are connected to at least $0.12n$ sources with distinct indices.*

Lemma 4. *For any $d \geq 4$, Chung's construction yields a d-regular bipartite graph that is an $(n, \alpha, (d/3)\alpha)$ bipartite expander for every $\alpha \leq \frac{3}{2d}$ with probability $1 - \mathsf{negl}(nH_b(\alpha))$.*

3 Stacked DRG Proof of Space

In this section we show that stacking DRGs with bipartite expander edges between layers yields an arbitrarily tight proof of space with the number of layers increasing as $O(\log_2(1/\epsilon))$ where ϵ is the desired space gap. Moreover, the proof size is also $O(\log_2(1/\epsilon))$, which is asymptotically optimal. Our proofs attempt a tight analysis as well, e.g. showing that just 10 layers achieve a PoS with a 1% space gap, degree $8 + d$ graphs where d is the degree of the DRG, and relies only on a DRG that retains depth in 80% subgraphs.

3.1 Review of the Stacked-Expander PoS

The PoS construction by Ren and Devadas [19] based on stacked bipartite expander graphs is a building block towards our tight PoS construction. Their construction uses a layered graph where each layer is a directed line on n nodes and the directed edges of a bipartite expander graph are placed between layers. This was shown to be an $(\epsilon\gamma n, (1-2\epsilon)\gamma n)$-sound PoS for parameters $\epsilon < 1/2$ and $\gamma < 1$ [19].

The graph $\mathcal{G}_{SE}$ The stacked-expander PoS uses the same underlying graph as the Balloon Hash memory hard function [12]. The graph $\mathcal{G}_{SE}$ consists of $\ell = O(\lambda)$ layers $V_1, ..., V_\ell$ consisting each of n vertices indexed in each level by the integers $[n]$, and where λ is a security parameter. First directed edges are placed from each kth vertex to the $k+1$st vertex in each level, i.e. forming a directed line. Next directed edges are placed from V_{i-1} to V_i according to the edges of an (n, α, β) bipartite expander on (V_{i-1}, V_i). Finally a “localization” operation is applied so that each kth vertex u_k in V_{i-1} is connected to the kth vertex v_k in V_i and any directed edge from the kth vertex of V_{i-1} to some jth vertex of V_i where $j > k$ is replaced with a directed edge from the kth vertex of V_i to the jth vertex of V_i. $\mathcal{G}_{SE}$ can be pebbled in $n\ell$ steps using a total of n pebbles.

Stacked-expander PoS The PoS follows the generic PoS based on graph labeling. We remark only on several nuances. Due to the topology of $\mathcal{G}_{SE}$ after localization, the prover only needs to use a buffer of size n and deletes the labels of V_{i-1} as it derives the labels of V_i. After completing the labels C_i in the ith level it computes a vector commitment (e.g. Merkle commitment) to the labels in C_i denoted com_i. Once it has derived the labels C_ℓ of the final level V_ℓ it computes $com = H_{id}(com_1 || \cdots || com_\ell)$ and uses $H_{id}(com||j)$ to derive λ non-interactive challenges for each jth level.

3.2 A new tight PoS from stacked DRGs

By simply replacing each of the path graphs V_i in the stacked-expander PoS construction with a depth robust graph results in an arbitrarily tight PoS. Specifically, only $O(\log 1/\epsilon)$ layers are needed to achieve a $((1-\epsilon)n, \Omega(n))$-parallel-sound PoS. We demand only very basic properties from the DRG, e.g. that any subgraph on 80% of the nodes contains a long path of $\Omega(n)$ length.

Construction of $\boldsymbol{\mathcal{G}_{SDR}[\ell]}$ The graph $\mathcal{G}_{SDR}[\ell]$ will be exactly like $\mathcal{G}_{SE}$ only each of the ℓ layers $V_1, ..., V_\ell$ contains a copy of an $(n, 0.80n, \beta n)$-depth-robust graph for some constant β. For concreteness, we define the directed edges between the layers using the degree 8 Chung random bipartite graph construction. For simplicity we will analyze the construction without applying localization to the expander edges between layers. Even without localization this is already a valid PoS, only the initialization requires a buffer of size $2n$ rather than n. The PoS is still "tight" with respect to the persistent space storage.

Vector commitment storage If the vector commitment storage overhead required for the PoS is significant then this somewhat defeats the point of a tight PoS. Luckily this is not the case. Most vector commitment protocols, including the standard Merkle tree, offer smooth time/space tradeoffs. With a Merkle tree the honest prover can delete the hashes on nodes on the first k levels of the tree to save a factor 2^k space and re-derive all hashes along a Merkle path by reading at most 2^k nodes and computing at most 2^k hashes. If $k = 7$ this is less than a 1% overhead in space, and requires at most 128 additional hashes and reads. Furthermore, as remarked in [18] these 2^k reads are sequential memory reads, which in practice are inexpensive compared to the random reads for challenge labels.

Proof size We show that $\ell = O(\log(\frac{1}{3(\epsilon-2\delta)}))$ suffices to achieve $\mathsf{negl}(\lambda)$ soundness against any prover running in parallel time less than βn rounds of queries where Chal samples λ/δ nodes in each layer. This would result in a proof size of $O((1/\epsilon)\log(1/\epsilon))$, which is already a major improvement on any PoS involving a graph of degree $O(1/\epsilon)$ (recall that the only previously known tight PoS construction relied on very special DRGs whose degree must scale with $1/\epsilon$, which results in a total proof size of $O(1/\epsilon^2)$). However, we are able to improve the result even further and show that only $O(1/\delta)$ challenge queries are required overall, achieving proof complexity $O(1/\epsilon)$. This is the optimal proof complexity for the generic pebbling-based PoS with at most an ϵ space gap. If the prover claims to be storing n pebbles and the proof queries less than $1/\epsilon$ then a random deletion of an ϵ fraction of these pebbles evades detection with probability at least $(1-\epsilon)^{1/\epsilon} \approx 1/e$. The same applies if a random ϵ fraction of the pebbles the prover claims to be storing are red (i.e. errors).

Analysis outline We prove the hardness of the red-black pebbling game $\mathsf{Red\text{-}Black\text{-}Pebbles}^{\mathcal{A}}(\mathcal{G}_{SDR}[\ell], V_\ell, \mathsf{Chal})$ where Chal samples λ_i uniform challenges over V_i. We first show that it suffices to consider the parallel complexity of pebbling the set $U_\ell \subseteq V_\ell$ of *all* unpebbled nodes on V_ℓ from an initial configuration of γn black pebbles overall and $\delta_i n$ red pebbles in each layer where $\delta_\ell < \epsilon/2$.

As a shorthand notation, we will say that $\mathcal{G}_{SDR}[\ell]$ is $(\gamma, \boldsymbol{\delta}, t, \mu)$-hard if every subset containing a μ fraction of the nodes in V_ℓ require t rounds to pebble (i.e. greater than a $1-\mu$ fraction of the nodes each individually require t rounds to pebble) from an initial configuration of γn black pebbles overall and $\delta_i n$ red pebbles in each layer where $\delta_\ell < \epsilon/2$. In Claim 1 we show that if $\mathcal{G}_{SDR}[\ell]$ is $(\gamma, \boldsymbol{\delta}, t, \mu)$-hard then the labeling PoS on $\mathcal{G}_{SDR}[\ell]$ is $(\gamma n, t, max\{p^*, \mu^\kappa\})$-sound where $p^* = max_i(1-\delta_i)^{\lambda_i}$. (Recall that κ and $\boldsymbol{\lambda}$ are parameters defined in the game).

For $\mu = 1$, $(\gamma, \boldsymbol{\delta}, t, 1)$-hardness is nearly equivalent to the standard parallel pebbling complexity of U_ℓ. The one distinction[5] is its dependency on the restriction to δ_i red pebbles in each layer, counted separately from black pebbles. In Claim 3 we show that if $\mathcal{G}_{SDR}[\ell]$ is $(1-\epsilon+2\delta_\ell, \boldsymbol{\delta}, t, 1)$-hard then $\mathcal{G}_{SDR}[\ell+1]$ is $(1-\epsilon, \boldsymbol{\delta}^*, t, 1-\epsilon/2)$-hard where $\boldsymbol{\delta}^*$ is equal to $\boldsymbol{\delta}$ on all common indices and $\delta_{\ell+1} = \delta_\ell$.

Finally, we analyze the complexity of pebbling all of U_ℓ, i.e. the $(\gamma, \boldsymbol{\delta}, t, 1)$-hardness of $\mathcal{G}_{SDR}[\ell]$. We show in Claim 5 that when the adversary uses at most $\gamma < 1-\epsilon$ black pebbles and δ red pebbles in each layer then pebbling all the unpebbled nodes in layer V_ℓ (for ℓ dependent on ϵ and δ) requires pebbling $0.80n$ unpebbled nodes (including both red and black pebbles) in some layer V_i. Since the layer V_i contains a $(n, 0.80, \beta n)$-depth-robust graph, this takes at least βn rounds. We then generalize this analysis (Claim 6) to apply when δ_i is allowed to increase from level ℓ to 1 by a multiplicative factor such that $\sum_i \delta_i = O(\delta_\ell)$.

[5] In prior uses of the red-black pebbling game to analyze proofs of space, it sufficed to consider parallel black pebbling complexity because replacing red pebbles with "free" black pebbles only increases the adversary's power. Our more refined analysis requires analyzing the weaker adversary who is restricted to a maximum number of red pebbles on each level of the graph, enforced by the construction.

Theorem 2 ties everything together, taking into account the constraints of each claim to derive the PoS soundness of the labeling PoS on $\mathcal{G}_{SDR}[\ell]$.

Theorem 2. *The labeling PoS on $\mathcal{G}_{SDR}[\ell]$ with Chal sampling λ_i challenges in each level V_i and κ online challenges in V_ℓ is $((1-\epsilon-\delta)n, \beta n, e^{-\lambda})$-sound with $\kappa = 2\lambda/\epsilon$ if either of the following conditions are met for $\epsilon \leq 0.24$:*

(a) $\ell = max(8, \log_2(\frac{1}{3(\epsilon-2\delta)})+4)$ *and each* $\lambda_i = \lambda/\delta$ *and* $\delta < min(0.01, \epsilon/3)$

(b) $\ell = max(14, \log_2(\frac{1}{3(\epsilon-3\delta)}) + 5)$ *and each* $\lambda_i = \lambda/\delta_i$ *where* $\delta_\ell = \delta < min(0.01, \epsilon/2)$ *and* $\delta_i = min(0.05, \frac{2}{3}\delta_{i-1})$

Proof. For any $\mathcal{G}_{SDR}[\ell]$, if the set of unpebbled nodes in V_ℓ are connected via unpebbled paths to at least $0.80n$ unpebbled nodes (including red and black) in some prior level V_i, then pebbling all of V_ℓ requires pebbling all these $0.80n$ unpebbled nodes, which requires βn rounds due to the fact that V_i is $(n, 0.80n, \beta n)$ depth robust. Claim 5 implies that $\mathcal{G}_{SDR}[\ell]$ is $(1-\epsilon, \boldsymbol{\delta}, \beta n, 1)$-hard for $\boldsymbol{\delta}$ and ℓ such that $\delta_i = \delta < \epsilon/2$ for all i and $\ell = max(7, \log_2(\frac{1}{3(\epsilon-2\delta)} + 3))$.

Claim 6 gives a different tradeoff between ℓ and $\boldsymbol{\delta}$, showing that the same hardness holds for $\boldsymbol{\delta}$ and ℓ such that $\delta_\ell = \delta < \epsilon/3$ and $\delta_i = min(0.05, \frac{2}{3}\delta_{i-1})$ and $\ell = max(13, \log_2(\frac{1}{3(\epsilon-3\delta)}) + 4)$.

Assuming $\epsilon \leq 0.24$, Claim 3 implies that $\mathcal{G}_{SDR}[\ell+1]$ is $(1-\epsilon-\delta, \boldsymbol{\delta}, \beta n, 1-\epsilon/2)$-hard extending $\boldsymbol{\delta}$ so that $\delta_{\ell+1} = \delta_\ell = \delta$. Finally, by Claim 1, the labeling PoS on $\mathcal{G}_{SDR}[\ell+1]$ with challenge set V_ℓ and Chal sampling λ_i in each level V_i is $((1-\epsilon-\delta)n, \beta n, max\{p^*, (1-\epsilon/2)^\kappa\})$-sound where $p* = max_i(1-\delta_i)^{\lambda_i}$. Setting $\lambda_i = \lambda/\delta_i$ and $\kappa = 2\lambda/\epsilon$, the PoS is $((1-\epsilon-\delta)n, \beta n, e^{-\lambda})$-sound.

Notation 1 (Common analysis notations) *Let U_i denote the entire index set of nodes that are unpebbled in V_i and P_i the set that are pebbled. The total number of pebbles placed in the initial configuration is γn. Each level initially has $\rho_i n$ black pebbles and $\delta_i n$ red pebbles. Finally, $\gamma_i n = \sum_{j<i} \rho_i n$ is the number of black pebbles placed before level i.*

Claim 1 *If $\mathcal{G}_{SDR}[\ell]$ is $(\gamma, \boldsymbol{\delta}, t, \mu)$-hard then the labeling PoS on $\mathcal{G}_{SDR}[\ell]$ is $(\gamma n, t, max\{p^*, \mu^\kappa\})$-sound where $p^* = max_i(1-\delta_i)^{\lambda_i}$.*

Proof. Fix $\gamma = 1-\epsilon$ and δ_i for each i. The λ_i challenges during Initialization in each level ensure that $\mathcal{A}$ wins with at most probability $(1-\delta_i)^{\lambda_i}$ if it places more than δ_i red pebbles on V_i. If $\mathcal{A}$ has exceeded the δ_i bound in more than one level this only increases its probability of failure. Thus, in case 1 ($\mathcal{A}$ places more than δ_i red pebbles on some level i), $\mathcal{A}$'s success probability is bounded by maximum value of $(1-\delta_i)^{\lambda_i}$ over all i. In case 2 ($\mathcal{A}$ places fewer than δ_i on each ith level), the fact that $\mathcal{G}_{SDR}[\ell]$ is $(\gamma, \boldsymbol{\delta}, t, \mu)$-hard implies that at most a μ fraction of the nodes on V_ℓ can individually be pebbled from the starting configuration in t rounds, hence $\mathcal{A}$'s success probability of answering κ independent challenges is bounded by μ^κ. The success probability is bounded by the maximum of these two cases.

Claim 2 (trivial) *$\mathcal{G}_{SDR}[\ell]$ is $(\gamma, \boldsymbol{\delta}, t, 1)$-hard if and only if given any initial configuration P_0 of $\gamma_\ell n$ black pebbles placed on layers $V_1, ..., V_{\ell-1}$ at most $\delta_i n$ red pebbles in each layer, and any set $U \subseteq V_\ell$ of αn unpebbled nodes in V_ℓ such that $\alpha - \gamma_\ell \geq 1 - \gamma - \delta$, no adversary can pebble U in fewer than t rounds.*

The proof of this claim is in the full version of the paper.

Claim 3 *For any $\epsilon \leq 0.24$, if $\mathcal{G}_{SDR}[\ell-1]$ is $(1-\epsilon+\delta_{\ell-1}, \boldsymbol{\delta}, t-1, 1)$-hard then $\mathcal{G}_{SDR}[\ell]$ is $(1-\epsilon, \boldsymbol{\delta}^*, min(\beta n, t), 1-\epsilon/2)$-hard where $\boldsymbol{\delta}^*$ is identical to $\boldsymbol{v}$ on all common indices and $\delta_\ell = \delta_{\ell-1} \leq \epsilon/2$.*

Proof. Refer to Notation 1. Consider the graph $\mathcal{G}_{SDR}[\ell]$ with $\gamma n = (1-\epsilon)n$ black pebbles initially placed. Let $\delta = \delta_\ell = \delta_{\ell-1} \leq \epsilon/2$. Let $\alpha_\ell = |U_\ell|/|V_\ell|$ denote the fraction of nodes in V_ℓ that are unpebbled. Every $1-\epsilon/2$ fraction of V_ℓ contains at least $\alpha^* n = (\alpha_\ell - \epsilon/2)n$ unpebbled nodes. If $\alpha^* \geq 0.80$, then every $1-\epsilon/2$ fraction of V_ℓ contains a path of length βn because V_ℓ is a $(n, 0.80n, \beta n)$-depth robust graph. We consider the two other cases next:

Case $\alpha^* < 1/3$: The $\alpha^* n$ unpebbled nodes have dependencies on at least a $2\alpha^* = 2\alpha_\ell - \epsilon$ fraction of nodes in $V_{\ell-1}$ (Corollary 1, bipartite expansion). These contain at least $\alpha' n = (2\alpha_\ell - \epsilon - \rho_{\ell-1} - \delta)n$ unpebbled nodes because in the worst case they include $\rho_{\ell-1}n$ black pebbles and δn red pebbles. There are $\gamma_{\ell-1}n = (\gamma - \rho_{\ell-1} - \rho_\ell)n$ pebbles placed on all prior levels. By definition $\rho_\ell = 1 - \alpha_\ell - \delta$ and $\gamma = 1 - \epsilon$. Substituting $\alpha_\ell \geq 1-\gamma-\delta$ shows that $\alpha' - \gamma_{\ell-1} = 2\alpha_\ell - \gamma + \rho_\ell - \epsilon - \delta = \alpha_\ell + 1 - \gamma - \epsilon - 2\delta \geq 1 - 2\gamma - 3\delta + 1 - \epsilon = 1 - \gamma - 3\delta$. Setting $\gamma' = 1 - \epsilon + \delta = \gamma + \delta$ gives the relation $\alpha' - \gamma_{\ell-1} \geq 1 - \gamma' - \delta$. It then follows from Claim 2 that if $\mathcal{G}_{SDR}[\ell-1]$ is $(\gamma', \boldsymbol{\delta}, t-1, 1)$-hard then the $\alpha' n$ unpebbled nodes in $V_{\ell-1}$

require $t-1$ rounds to pebbled. Thus, the $\alpha^* n$ unpebbled nodes in V_ℓ require t rounds to pebble.

Case $\alpha^* \geq 1/3$: In this case $\alpha^* \in (0.33, 0.80)$. It is connected to $\beta^* n$ nodes in $V_{\ell-1}$. Among these at least $\alpha' n$ for $\alpha' \geq \beta^* - \rho_{\ell-1} - \delta$ are unpebbled. Since $\gamma_{\ell-1} = \gamma - \rho_\ell - \rho_{\ell-1}$ we get $\alpha' - \gamma_{\ell-1} \geq \beta^* - \gamma + \rho_\ell - \delta$. According to Corollary 2 on the bipartite expander boundary $\beta^* - \alpha^* \geq 0.12$. Therefore, $\rho_\ell = 1 - \alpha_\ell - \delta \geq 1 - \alpha^* - \epsilon/2 - \delta$ so $\alpha' - \gamma_{\ell-1} \geq \beta^* - \gamma + 1 - \alpha^* - \epsilon/2 - 2\delta = \beta^* - \alpha^* + \epsilon/2 - 2\delta \geq 0.12 + \epsilon/2 - 2\delta$. If $\mathcal{G}_{SDR}[\ell-1]$ is $(1-\epsilon+\delta, \boldsymbol{\delta}, t-1, 1)$-hard, then by Claim 2 the $\alpha' n$ unpebbled nodes require $t-1$ rounds as long as $0.12 + \epsilon/2 - 2\delta \geq \epsilon - 2\delta$, which is true for $\epsilon \leq 0.24$.

Claim 4 *If $\mathcal{G}_{SDR}$ initially has at most γn black pebbles for $\gamma \leq 1-\epsilon$ and at most $\delta n < \epsilon n/2$ red pebbles in each layer then for $\ell = \log_2(\frac{1}{3(\epsilon-2\delta)})$ the unpebbled nodes in V_ℓ have unpebbled paths from at least $n/3$ unpebbled nodes in some layer V_i.*

Proof. Refer to Notations 1. Let $\alpha_i n$ denote the number of unpebbled dependencies of U_ℓ in V_i, i.e. the number of nodes in U_i that have unpebbled paths to U_ℓ. Suppose that α_i is bounded by $1/3$ for all levels up to $\ell - k$, i.e. $\alpha_\ell < 1/3, ..., \alpha_{\ell-k} < 1/3$. We will prove the following bound:

$$\alpha_{\ell-k} \geq 2^k(\alpha_\ell - \gamma_\ell/2 - \delta) \geq 2^{k-1}(\alpha_\ell + \epsilon - 3\delta) \geq 2^k(\epsilon - 2\delta) \qquad (3.1)$$

Before proving this bound let us note its implication. For $k = \log_2(\frac{1}{3(\epsilon-2\delta)})$ this implies $\alpha_{\ell-k} \geq 1/3$, which contradicts $\alpha_{\ell-k} < 1/3$. Therefore, it follows that $\alpha_i \geq 1/3$ at some index $i > \ell - \log_2(\frac{1}{3(\epsilon-2\delta)})$, which leads to the conclusion that for $\ell \geq \log_2(\frac{1}{3(\epsilon-2\delta)})$ there exists some level V_i with at least $n/3$ unpebbled nodes that have unpebbled dependency paths to the set X_ℓ of unpebbled nodes in V_ℓ.

Let $j = \ell - i$. From Corollary 1 (bipartite expansion), if $\alpha_j \leq 1/3$ then X_j is connected to at least $2\alpha_j$ nodes in V_{j-1}. At most $(\rho_{j-1} + \delta)n$ of these are pebbled. Therefore, $\alpha_{j-1} \geq 2\alpha_j - \rho_{j-1} - \delta$. Now we show by induction that $\alpha_{\ell-k} \geq 2^k\alpha_\ell - 2^{k-1}\rho_{\ell-1} - (2^k-1)\delta$. The base case $k=0$ is trivial. Assuming this holds for k:

$$\begin{aligned}\alpha_{\ell-k-1} &\geq 2\alpha_{\ell-k} - \rho_{\ell-k-1} - \delta \geq 2(2^k\alpha_\ell - 2^{k-1}\rho_{\ell-1} - (2^k-1)\delta) - \rho_{\ell-k-1} - \delta \\ &\geq 2^{k+1}\alpha_\ell - 2^k\rho_{\ell-1} - (2^{k+1}-1)\delta\end{aligned}$$

The last inequality used the fact that $\sum_{i=1}^k \rho_{\ell-i} \leq \gamma_\ell$ and therefore $\sum_{i=1}^k 2^{k-i}\rho_{\ell-i}$ is maximized by setting $\rho_{\ell-1} = \gamma_\ell$ and $\rho_{\ell-i} = 0$ for all $i > 1$.

From the identities $\gamma_\ell = \gamma - \rho_\ell$ and $\alpha_\ell = 1 - \rho_\ell - \delta$ we derive $\gamma_\ell = \gamma + \alpha_\ell - 1 + \delta \leq \alpha_\ell + \delta - \epsilon$. Finally, inserting this into the bound above and using the fact that $\alpha_\ell \geq \epsilon - \delta$ gives:

$$\alpha_{\ell-k} \geq 2^{k-1}(2\alpha_\ell - \gamma_\ell - 2\delta) \geq 2^{k-1}(\alpha_\ell + \epsilon - 3\delta) \geq 2^k(\epsilon - 2\delta)$$

We could stop here as we have already shown unpebbled dependency paths from the unpebbled sinks in V_ℓ to a $1/3$ fraction of nodes in some level for $\ell = O(\log(1/(\epsilon - 2\delta))$ and the remainder of our PoS analysis could rely on a graph that is $(n, 0.33n, \Omega(n))$-depth-robust. However, we can tighten the analysis further so that we only need to assume the graph is $(n, 0.80, \Omega(n))$-depth robust.

Claim 5 *If $\mathcal{G}_{SDR}$ initially has at most γn black pebbles for $\gamma \leq 1 - \epsilon$ and at most $\delta < \epsilon/2$ red pebbles in each layer then the unpebbled nodes in V_ℓ have unpebbled paths to at least $0.80n$ unpebbled nodes in some layer V_i for $\ell = max(\frac{0.68-\epsilon+\delta}{0.12-\delta}, \log_2(\frac{1}{3(\epsilon-2\delta)})) + 3)$. In particular, $\ell = max(7, \log_2(\frac{1}{3(\epsilon-2\delta)})) + 3)$ when $\delta \leq 0.01$.*

Proof. In Claim 4 we showed that for $\ell \geq \log_2(\frac{2}{3(\alpha_\ell+\epsilon-3\delta)})$ there exists an index i where $\alpha_i \geq 1/3$ and $\alpha_\ell + \epsilon - 3\delta \geq 2\epsilon - 4\delta$ (Equation 3.1). Picking up from here, we consider what happens once $\alpha_i \geq 1/3$. We break the analysis into two cases: in the first case $\alpha_\ell < 1/3$ and in the second case $\alpha_\ell \geq 1/3$.

In both cases we will use a different bound on α_{i-k} because once $\alpha_i > 1/3$ the unpebbled sets may not be 2-expanding. Define the function $\beta(\alpha)$ to be the minimum bipartite expansion of a set of fractional size α, i.e. every set of αn nodes is connected to at least $\beta(\alpha)n$ nodes in the previous level. Let $\hat{\beta}(\alpha) = \beta(\alpha) - \alpha$. Using the relation $\alpha_{i-1} \geq \beta(\alpha_i) - \rho_{i-1} - \delta$ we derive that $\alpha_{i-2} \geq \hat{\beta}(\alpha_{i-1}) + \beta(\alpha_i) - \rho_{i-1} - \rho_{i-2} - 2\delta$ and more generally, since $\sum_{j=1}^{k} \rho_{i-j} \leq \gamma_i$:

$$\begin{aligned}\alpha_{i-k} &\geq \sum_{j=1}^{k-1} \hat{\beta}(\alpha_{i-j}) + \beta(\alpha_i) - k\delta - \sum_{j=1}^{k} \rho_{i-j} \\ &\geq (k-1)(min_{j<k}\hat{\beta}(\alpha_{i-j}) - \delta) + \beta(\alpha_i) - \gamma_i - \delta\end{aligned}$$

The final ingredient is the bound $\gamma_i \leq \alpha_i - \epsilon + \delta$ for all i. To see this, first observe that $\alpha_\ell - \gamma_\ell = 1 - \rho_\ell - \delta - (\gamma - \rho_\ell) = \epsilon - \delta$. If $\alpha_i - \gamma_i \geq \epsilon - \delta$ and $\epsilon > 2\delta$, then $0.80 > \alpha_i > \delta$, and so the $\alpha_i n$ dependencies are connected to $\beta(\alpha_i)n > (\alpha_i + \delta)n$ nodes in level V_{i-1}. Therefore $\alpha_{i-1} \geq \alpha_i - \rho_{i-1} =$

$\alpha_i - (\gamma_i - \gamma_{i-1})$. In words, decreasing the number of dependencies requires using black pebbles 1-to-1, so $\alpha_{i-1} - \gamma_{i-1} \geq \alpha_i - \gamma_i$.

$$\alpha_{i-k} \geq (k-1)(min_{j<k}\hat{\beta}(\alpha_{i-j}) - \delta) + \hat{\beta}(\alpha_i) + \epsilon - 2\delta \tag{3.2}$$

By Corollary 2 to Lemma 3, $\hat{\beta}(\alpha) \geq 0.12$ for $\alpha \in (0.10, 0.80)$.
Case $\alpha_\ell \geq 1/3$: First we claim that $\alpha_{\ell-i} \geq 0.12$ for all i. If $\alpha_i \geq 0.12$ then as shown above $\alpha_{i-1} \geq \hat{\beta}(\alpha_i) + \epsilon - 2\delta \geq 0.12$ because $\hat{\beta}(\alpha) \geq 0.12$ for all $\alpha \in (0.10, 0.80)$ and $\epsilon > 2\delta$. Our claim thus follows by induction. Therefore, for all $j \leq k$ we derive that $min_{j\leq k}(\hat{\beta}(\alpha_{\ell-j})) \geq 0.12$. Equation 3.2 then shows that $\alpha_{\ell-k-1} \geq k(0.12-\delta) + 0.12 + \epsilon - \delta$, or $\alpha_{\ell-k-1} \geq 0.80$ at $k \geq (0.68 - \epsilon + \delta)/(0.12-\delta)$ (e.g. $k = 7$ when $\delta \leq 0.01$).

Case $\alpha_\ell < 1/3$: From Equation 3.1, $\alpha_i \geq 1/3$ at some index $i \geq \ell - k$ for $k = \log_2(\frac{2}{3(\alpha_\ell+\epsilon-3\delta)})$. At this point $\alpha_i \geq 1/3$ and $\gamma_i < \gamma_\ell \leq \alpha_\ell - \epsilon + \delta$. Combining this with Equation 3.2, we can apply the same analysis as in the previous case to first show by induction that $\alpha_{i-k'} \geq 0.12$ for all k' and then more generally: $\alpha_{i-k'} \geq (k'-1)(0.12-\delta) + \beta(\alpha_i) - \alpha_\ell + \epsilon - 2\delta \geq k'(0.12-\delta) + 0.68 - \alpha_\ell + \epsilon - 2\delta$. We used the fact that $\beta(\alpha_i) \geq \beta(0.33) \geq 0.68$. Therefore, $\alpha_{i-k'-1} \geq 0.80$ when $k' \geq (0.80 - 0.68 + \alpha_\ell)/(0.12-\delta)$. This shows that the total number of levels where $\alpha_i < 0.80$ is at most:

$$\ell = k + k' + 1 \leq 1 + \log_2\big(\frac{2}{3(\alpha_\ell + \epsilon - 3\delta)}\big) + \frac{0.12 + \alpha_\ell}{0.12 - \delta}$$

The derivative of this expression with respect to α_ℓ is $\frac{1}{0.12-\delta} - \frac{1}{\ln(2)(\alpha_\ell+\epsilon-3\delta)}$, which is initially decreasing when $\ln(2)(\alpha_\ell+\epsilon-3\delta) < 0.12-\delta$ and then increasing for larger α_ℓ. Therefore, the maxima are on the endpoints of the interval $\alpha_\ell \in (\epsilon-\delta, 0.33)$. We already considered the case $\alpha_\ell = 0.33$. When $\alpha_\ell = \epsilon - \delta$ then the number of levels is at most $1 - \log_2(3(\epsilon - 2\delta)) + \frac{0.12}{0.12-\delta}$.

In conclusion, the total number of levels before $\alpha_i \geq 0.80$ is at most:

$$\ell \leq max\big((0.68 - \epsilon + \delta)/(0.12 - \delta), 1 - \log_2\big(3(\epsilon - 2\delta)\big) + 1/(1 - \delta/12)\big)$$

In particular, when $\delta \leq 0.01$ this becomes $max(7, -\log_2(3(\epsilon-2\delta))+3)$.

Relaxing $\boldsymbol{\delta}$ Claim 5 improved on Claim 4 to show unpebbled dependency paths to 80% of the subgraph in some layer. The final improvement is to redistribute the δ_i such that $\sum_i \delta_i = O(\delta)$ but security is still maintained. Intuitively, ensuring $\delta < \epsilon$ is necessary on level V_ℓ as otherwise $\gamma + \delta \geq 1$ and there are no unpebbled nodes on level V_ℓ (all the missing black pebbles can be covered with red pebbles). However, as the dependencies expand between levels a larger δ can be tolerated as well. Although the number of black pebbles the prover will place on each level isn't fixed a priori, we show that if $\delta < \epsilon/2$ in level V_ℓ then we can tolerate a factor 3/2 increase between levels as long as $\delta \leq 0.05$ in any layer.

That is, if δ_i denotes the bound on the number of red pebbles in the ith layer then our new analysis requires $\delta_\ell < \epsilon/2$ and $\delta_i = min(0.05, (3/2)\delta_{i+1})$. This means that the total number of queries in the PoS over all levels is $O(1/\epsilon)$ because $\sum_{i=1}^{\ell} 1/\delta_i \leq max(0.10\ell, \frac{3}{2\delta_\ell})$.

Claim 6 *For any $\gamma \leq 1-\epsilon$ and $\delta < \epsilon/3$, if $\mathcal{G}_{SDR}$ initially has at most γn black pebbles, $\delta_\ell = \delta$ red pebbles in layer V_ℓ, and $\delta_i = min(0.05, (2/3)\delta_{i-1})$ red pebbles in layer V_i, then for $\ell = max(13, \log_2(\frac{1}{3(\epsilon-3\delta)})) + 4)$ the unpebbled nodes in V_ℓ have unpebbled paths to at least $0.80n$ unpebbled nodes in some layer V_i.*

Proof. Modifying Equation 3.1 to account for the different values of δ_i gives:

$$\begin{aligned}\alpha_{\ell-k} &\geq 2^k\alpha_\ell - 2^{k-1}\gamma_\ell - (2^{k-1}\delta_{\ell-1} + 2^{k-2}\delta_{\ell-2} + \cdots + \delta_{\ell-k}) \\ &\geq 2^k\alpha_\ell - 2^{k-1}\gamma_\ell - \sum_{i=1}^{k} 2^{k-i}(3/2)^{i-1}\delta_\ell\end{aligned}$$

Let $\sigma_k = \sum_{i=1}^{k} 2^{k-i}(3/2)^{i-1}$. Then $(4/3)\sigma_k = \sigma_k + 2^{k+1}/3 - (3/2)^{k-1}$. Therefore $\sigma_k = 2^{k+1} - 3^k/2^{k-1} < 2^{k+1}$. Using $\gamma_\ell \leq \alpha_\ell + \delta_\ell - \epsilon$ and $\alpha_\ell \geq \epsilon - \delta_\ell$ we derive the new bound:

$$\alpha_{\ell-k} \geq 2^k\alpha_\ell - 2^{k-1}\gamma_\ell - 2^{k+1}\delta_\ell \geq 2^{k-1}(\alpha_\ell + \epsilon - 5\delta_\ell) \geq 2^k(\epsilon - 3\delta) \quad (3.3)$$

This shows that if $\ell \geq \log_2(\frac{1}{3(\epsilon-3\delta)})$ then there is some level V_i where $\alpha_i \geq 1/3$.

We must also modify Equation 3.2 using $\sum_{j=1}^{k} \rho_{i-j} \leq \gamma_i \leq \alpha_i - \epsilon + \delta_i$:

$$\alpha_{i-k} \geq (k-1)min_{j<k}\hat{\beta}(\alpha_{i-j}) - \sum_{j=0}^{k} \delta_{i-j} + \hat{\beta}(\alpha_i) + \epsilon \quad (3.4)$$

When $i = \ell$ and k is small $\delta_{\ell-k} = (3/2)^k\delta_\ell$ and $\delta_\ell \leq \epsilon/3$ implies:

$$\alpha_{\ell-k} \geq (k-1)min_{j<k}\hat{\beta}(\alpha_{i-j}) + \hat{\beta}(\alpha_\ell) + (\frac{2}{3} - \frac{3^k}{2^{k+1}})\epsilon$$

Otherwise, we can use $\delta_i \leq 0.05$.

$$\alpha_{i-k} \geq (k-1)(min_{j<k}\hat{\beta}(\alpha_{i-j}) - 0.05) + \hat{\beta}(\alpha_i) + \epsilon - 0.10$$

Now we turn back to the two cases for $\alpha_i \geq 1/3$.

Case $\alpha_\ell \geq 1/3$*:* We claim that $\alpha_{\ell-k} \geq 0.11$ for all k. This is true for α_ℓ by hypothesis. From the equation above and the bound $\hat{\beta}(\alpha) \geq 0.12$ for all $\alpha \in (0.10, 0.80)$ (Corollary 2), $\alpha_{\ell-1} \geq \hat{\beta}(\alpha_\ell) - \epsilon/12 > 0.11$. Therefore, $\alpha_{\ell-2} \geq (0.12 - 0.05) + 0.12 - (11/24)\epsilon \geq 0.18$. Now assume that $\alpha_{\ell-2-j} \geq 0.12$ for all $j < k$, then $\alpha_{\ell-2-j} \geq (k-1)0.07 + 0.12 - (11/24)\epsilon > 0.11$. The claim follows by induction. This also shows that $\alpha_{\ell-k} \geq (k-3)0.07 + 0.11 > 0.80$ when $k = 13$.

Case $\alpha_\ell < 1/3$*:* From Equation 3.3, $\alpha_i \geq 1/3$ at some index $i \geq \ell - k$ for $k = \log_2(\frac{2}{3(\alpha_\ell+\epsilon-5\delta_\ell)})$. At this point $\alpha_i \geq 1/3$ and $\gamma_i < \gamma_\ell \leq \alpha_\ell - \epsilon + \delta_\ell$. Combining this with Equation 3.4 gives:

$$\alpha_{i-k'} \geq (k'-1)(min_{j<k'}\hat{\beta}(\alpha_{i-j}) - 0.05) + \beta(\alpha_i) - \alpha_\ell + \epsilon - 0.05 - \delta_\ell$$

We claim that $\alpha_{i-k'} \geq 0.30$ for all k'. Observe that $\alpha i - 1 \geq \beta(\alpha_i) - \alpha_\ell + \epsilon - 0.05 - \delta_\ell \geq 0.68 - 0.38 + \epsilon - \delta_e ll \geq 0.30$ for any value $\alpha_\ell < 0.33$ because $\beta(\alpha_i) \geq \beta(0.33) \geq 0.68$. Assuming this is true for all α_{i-j} where $1 < j \leq k'$ implies $\alpha_{i-k'} \geq (k'-1)0.07+0.30 \geq 30$. Therefore, we can state more generally that $\alpha_{i-k'} \geq (k'-1)0.07 + 0.68 - \alpha_\ell$ and $\alpha_{i-k'-1} \geq 0.80$ when $k' = (0.12 + \alpha_\ell)/0.07$. The total number of levels where $\alpha_i < 0.80$ is thus at most:

$$k + k' + 1 \leq 1 - \log_2((3/2)(\alpha_\ell + \epsilon - 5\delta_\ell)) + 2 + \alpha_\ell/0.07$$

Differentiating this expression with respect to α_ℓ shows that the maxima over $\alpha_\ell \in (\epsilon - \delta_\ell, 0.33)$ are on the endpoints. The endpoint $\alpha_\ell = 0.33$ coincides with the case above. At the endpoint $\epsilon - \delta_\ell$ the number of levels is bounded by $3 - \log_2(3(\alpha_\ell + \epsilon - 5\delta)) + \epsilon/0.07$.

In conclusion, considering both cases, the total number of levels before $\alpha_i \geq 0.80$ is at most:

$$\ell \geq max(13, 3 - \log_2(3(\epsilon - 3\delta_\ell)) + \epsilon/0.07)$$

In particular, when $\epsilon \leq 0.07$ and $\delta_\ell = \delta$ then $\ell \leq max(12, 4 - \log_2(3(\epsilon - 3\delta))$.

4 "ZigZag" DRG PoS/PoRep

The Stacked-DRG PoS can be adapted into a PoRep which encodes input data D using the labels on the last level V_ℓ as encoding keys. However, decoding the data requires first re-computing the PoS labels, which is by design expensive.

The basic idea of the ZigZag PoS/PoRep is to layer DRGs so that each layer "encodes" the previous layer. The critical desired property to achieve is: if *all* the labels on a given level are available in memory then the labels can decoded in parallel. To achieve this, instead of adding edge dependencies between the layers, we add the edges of a constant degree expander graph in each layer so that every layer is both depth-robust and has high "expansion". Technically, the graph we construct in each layer is an expander as an undirected graph. As a DAG this means that the union of the dependencies and targets of any subset is large. By alternating the direction of the edges between layers, forming a "zig-zag", we are able to show that the dependencies between layers expand. Now the only edges between layers are between nodes at the same index, and the label on each node encodes the label on the node at the same index in the previous level. The dependencies used for keys are all contained in the same layer. Thus, the labels in any layer are sufficient to recover the labels in the preceding layer. Moreover, the decoding step can be done in parallel.

Without alternating the direction of the edges between layers this construction would fail to be a tight proof of space because the topologically last ϵn nodes in a layer would only depend on the topologically last ϵn nodes in the previous layer. Moreover, if the prover stores the labels on the topologically first $(1-\epsilon)n$ nodes it can quickly recover the labels on the topologically first $(1-\epsilon)n$ nodes in the preceding level, allowing it to recover the missing ϵn labels as well in parallel-time $O(\epsilon n)$.

Construction of $\mathcal{G}_{ZZ}[\ell]$ Similar to $\mathcal{G}_{SDE}$, the graph $\mathcal{G}_{ZZ}[\ell]$ contains a copy of an $(n, 0.80n, \beta n)$-depth-robust graph for some constant β in each of the ℓ layers $V_1, ..., V_\ell$. The nodes in each layer are indexed in $[n]$. Every odd layer overlays the edges of the DRG in the forward direction (edges go from lower to higher indices) while every even layer the edges of the DRG in the reverse direction (edges go from higher indices to lower indices).

Edges are added between same index nodes in adjacent layers (i.e. the ith node in layer V_k is connected to the ith node in layer V_{k+1} for all i, k). Next, the edges that were between layers in $\mathcal{G}_{SDE}[\ell]$ are projected into each layer of $G_{ZZ}[\ell]$ with the direction of each edge determined by the parity of the layer. We call these *expander edges* to distinguish[6] them from other edges. More precisely, if $G_{SDE}[\ell]$ has an edge from the ith node of a layer V_k to the jth node of layer V_{k+1} then $G_{ZZ}[\ell]$ has an edge between the ith node of V_{k+1} and the jth node of V_{k+1}. The direction of the edges added to V_{k+1} is from lower indices to higher indices when $k+1$ is odd, and from higher indices to lower indices when $k+1$ is even. (For concreteness in the analysis, the edges between layers in the reference graph $\mathcal{G}_{SDE}$ are assumed to be constructed using the degree 8 Chung random bipartite graph construction).

DAG encoding Instead of the standard DAG labeling, the ZigZag PoS/PoRep uses a DAG encoding scheme. It takes in a data file X on n blocks $x_1, ..., x_n$, a salt σ for a collision-resistant hash function $H : \{0,1\}^{md} \to \{0,1\}^m$, and a d-inregular DAG on n nodes together with its parent function $\mathsf{Parents}(i)$ which outputs the parent nodes of the ith node. It uses a randomized encoding scheme $\mathsf{Enc}, \mathsf{Dec}$ to derive the label c_i on each node as $\mathsf{Enc}(k_i, x_i)$ where $k_i \leftarrow H(\sigma || c_{v_1} || \cdots || c_{v_d})$ for $(v_1, ..., v_d) \leftarrow \mathsf{Parents}(i)$. This encoding scheme may be as simple as the identity function, or could use sequentially slow encoding for added delay.

4.1 PoS analysis of ZigZag PoRep

Invertible pebbling game The red-black pebbling game no longer entirely captures the PoS security of the ZigZag PoRep due to the involvement of the encoding scheme ($\mathsf{Enc}, \mathsf{Dec}$) in the labeling rather than purely a collision-resistant hash function. Most significantly, the labels are now invertible. In terms of the dependency graph of the labeling computation, the keys in each layer V_i still need to be computed in topological order, however the labels may either be derived by decoding labels in layer V_{i+1} or encoding labels in layer V_{i-1}. We modify the black pebbling game to capture invertibility of labels by coloring edges.

White & green colored edges White edges are "one-way streets" corresponding to edge dependencies involved in deriving keys via calls to the random oracle and are treated like normal pebbling game edges. Green edges are "two-way street", but still have a direction and different rules in either direction. If there is a directed green edge from u to v then a pebble can be placed on v if and only if u and all nodes with white

[6] The distinction between expander edges and all other edges is important in the analysis. In particular, the expander edges are between the same index nodes in every layer and differ only in their directionality.

edges to v have pebbles. A pebble can be placed on u if and only if v and all nodes *with white edges to* v have pebbles.

We still analyze the soundness of a PoS with invertible labels through the game $\mathsf{Red\text{-}Black\text{-}Pebbles}$ as in Definition 1, however with the modification that the adversary plays the black pebbling game with white/green edges instead of the plain black pebbling game. Specifically we analyze the hardness of a modification of $\mathsf{Red\text{-}Black\text{-}Pebbles}^{\mathcal{A}}(\mathcal{G}_{ZZ}[\ell], V_\ell, \mathsf{Chal})$ using green/white edges where the directed edges within every layer V_i are white and the directed edges between the same index nodes in adjacent layers are green. Our analysis, included in the full version of this paper, will demonstrate in this model that the ZigZag PoRep is an arbitrarily tight PoS with only $\ell = O(\log(1/\epsilon)$ layers.

Acknowledgments

This research was generously supported by an NSF Graduate Fellowship. Joseph Bonneau, Nicola Greco, and Juan Benet provided critical input throughout the development of this work and are coauthors on a related systems project prototyping practical implementations of PoReps, including the constructions presented at BPASE 2018 and discussed in further detail in this work. Many others have contributed through helpful comments and conversations, including Dan Boneh, Rafael Pass, Ethan Cecchetti, Benedikt Bünz, and Florian Tramer.

References

[1] Proof of replication. Protocol Labs, 2017. `https://filecoin.io/proof-of-replication.pdf`.

[2] Hamza Abusalah, Joël Alwen, Bram Cohen, Danylo Khilko, Krzysztof Pietrzak, and Leonid Reyzin. Beyond hellman's time-memory trade-offs with applications to proofs of space. In *ASIACRYPT*, 2017.

[3] Joël Alwen, Jeremiah Blocki, and Benjamin Harsha. Practical graphs for optimal side-channel resistant memory-hard functions. In *CCS*, 2017.

[4] Joël Alwen, Jeremiah Blocki, and Krzysztof Pietrzak. Sustained space complexity. In *EUROCRYPT*, 2018.

[5] Dan Boneh, Joseph Bonneau, Benedikt Bunz, and Ben Fisch. Verifiable delay functions. 2018. To appear in CRYPTO 2018.

[6] F.R.K. Chung. On concentrators, superconcentrators, generalizers, and nonblocking networks. In *Bell System Technical Journal*, 1979.

[7] Moni Naor Cynthia Dwork and Hoeteck Wee. Pebbling and proofs of work. In *CRYPTO*, 2005.

[8] Stefan Dziembowski, Sebastian Faust, Vladimir Kolmogorov, and Krzysztof Pietrzak. Proofs of space. In *CRYPTO*, 2015.

[9] Ben Fisch. Poreps: Proofs of space on useful data. Cryptology ePrint Archive, Report 2018/678, 2018. `https://eprint.iacr.org/2018/678`.

[10] Ben Fisch, Joseph Bonneau, Juan Benet, and Nicola Greco. Proofs of replication using depth robust graphs. In *Presentation at Blockchain Protocol Analysis and Security Engineering 2018*, 2018. `https://cyber.stanford.edu/bpase2018`.

[11] Antonio Faonio Giuseppe Ateniese, Ilario Bonacina and Nicola Galesi. Proofs of space: when space is of the essence. In *Security and Cryptography for Networks, 2014.*, 2014.

[12] Dan Boneh Henry Corrigan-Gibbs and Stuart Schechter. Balloon hashing: a provably memory-hard function with a data-independent access pattern. In *Asiacrypt*, 2016.

[13] Sergio Demian Lerner. Proof of unique blockchain storage, 2014. `https://bitslog.wordpress.com/2014/11/03/proof-of-local-blockchain-storage/`.

[14] Mohammad Mahmoody, Tal Moran, and Salil P Vadhan. Time-lock puzzles in the random oracle model. In *CRYPTO*. Springer, 2011.

[15] Salil Vadhan Omer Reingold and Avi Wigderson. Entropy waves, the zig-zag graph product, and new constant-degree expanders and extractors. In *FOCS*, 2000.

[16] Sunoo Park, Krzysztof Pietrzak, Albert Kwon, Joël Alwen, Georg Fuchsbauer, and Peter Gai. Spacemint: A cryptocurrency based on proofs of space. Cryptology ePrint Archive, Report 2015/528, 2015. `http://eprint.iacr.org/2015/528`.

[17] Ronald L. Graham Paul Erdös and Endre Szemeredi. On sparse graphs with dense long paths. In *Computers & Mathematics with Applications*, 1975.

[18] Krzysztof Pietrzak. Proofs of Catalytic Space. Cryptology ePrint Archive # 2018/194, 2018.

[19] Ling Ren and Srinivas Devadas. Proof of space from stacked expanders. In *TCC*, 2016.