Depth-Robust Graphs and Their Cumulative Memory Complexity

Joël Alwen[1], Jeremiah Blocki[2], and Krzysztof Pietrzak[1]

[1] IST Austria
[2] Purdue University

Abstract. Data-independent Memory Hard Functions (iMHFS) are finding a growing number of applications in security; especially in the domain of password hashing. An important property of a concrete iMHF is specified by fixing a directed acyclic graph (DAG) G_n on n nodes. The quality of that iMHF is then captured by the following two pebbling complexities of G_n:

- The parallel cumulative pebbling complexity $\Pi^{\parallel}_{cc}(G_n)$ must be as high as possible (to ensure that the amortized cost of computing the function on dedicated hardware is dominated by the cost of memory).
- The sequential space-time pebbling complexity $\Pi_{st}(G_n)$ should be as close as possible to $\Pi^{\parallel}_{cc}(G_n)$ (to ensure that using many cores in parallel and amortizing over many instances does not give much of an advantage).

In this paper we construct a family of DAGs with best possible parameters in an asymptotic sense, i.e., where $\Pi^{\parallel}_{cc}(G_n) = \Omega(n^2/\log(n))$ (which matches a known upper bound) and $\Pi_{st}(G_n)$ is within a constant factor of $\Pi^{\parallel}_{cc}(G_n)$.

Our analysis relies on a new connection between the pebbling complexity of a DAG and its depth-robustness (DR) – a well studied combinatorial property. We show that high DR is *sufficient* for high $\Pi^{\parallel}_{cc}$. Alwen and Blocki (CRYPTO'16) showed that high DR is *necessary* and so, together, these results fully characterize DAGs with high $\Pi^{\parallel}_{cc}$ in terms of DR.

Complementing these results, we provide new upper and lower bounds on the $\Pi^{\parallel}_{cc}$ of several important candidate iMHFs from the literature. We give the first lower bounds on the memory hardness of the Catena and Balloon Hashing functions in a parallel model of computation and we give the first lower bounds of any kind for (a version) of Argon2i.

Finally we describe a new class of pebbling attacks improving on those of Alwen and Blocki (CRYPTO'16). By instantiating these attacks we upperbound the $\Pi^{\parallel}_{cc}$ of the Password Hashing Competition winner Argon2i and one of the Balloon Hashing functions by $O\left(n^{1.71}\right)$. We also show an upper bound of $O(n^{1.625})$ for the Catena functions and the two remaining Balloon Hashing functions.

1 Introduction

Moderately hard functions. Functions which are "moderately" hard to compute have found a variety of practical applications including password hashing, key-derivation and for proofs of work. In the context of password hashing, the goal is to minimize the damage done by a security breach where an adversary learns the password file; Instead of storing $(login, password)$ tuples in the clear, one picks a random salt and stores a tuple $(login, f(password, salt), salt)$, where $f(.)$ is a moderately hard function $f(.)$. This comes at a price, the server verifying a password must evaluate $f(.)$, which thus cannot be too hard. On the other hand, if a tuple $(login, y, salt)$ is leaked, an adversary who tries to find the password by a dictionary attack must evaluate $f(.)$ for every attempt. A popular moderately hard function is PBKDF2 (Password Based Key Derivation Function 2) [Kal00], which basically just iterates a cryptographic hash function H several times (1024 is a typical value).

Unfortunately a moderately hard function like PBKDF2 offers much less protection against adversaries who can build customized hardware to evaluate the underlying hash function than one would hope for. The reason is that the cost of computing a hash function H like SHA256 or MD5 on an ASIC (Application Specific Integrated Circuit) is orders of magnitude smaller than the cost of computing H on traditional hardware [DGN03,NBF$^+$15].

Memory-Bound and Memory-Hard Functions. [ABW03] recognized that cache-misses are more egalitarian than computation, in the sense that they cost about the same on different architectures. They propose "memory-bound" functions, which are functions that will incur many expensive cache-misses. This idea was further developed by [DGN03].

Along similar lines, Percival [Per09] observes that unlike computation, memory costs tend to be relatively stable across different architectures, and suggests to use memory-hard functions (MHF) for password hashing. [Per09] also introduced the `scrypt` MHF which has found a variety of applications in practice. Very recently it has been proven to indeed offer optimal time/space trade-offs in the random oracle model [ACP$^+$17,ACK$^+$16].

MHFs come in two flavours, data-dependent MHFs (dMHF) such as `scrypt`, and data independent MHFs (iMHF). The former are potentially easier to construct and allow for more extreme memory-hardness [ACP$^+$17,AB16], but they leave open the possibility of side-channel attacks [FLW13], thus iMHFs are preferable when the inputs are sensitive, as in the case of password hashing. We shortly discuss the state of the art for dMHFs at the end of this section.

iMHF as Graphs. An iMHF comes with an algorithm that computes the function using a fixed memory access pattern. In particular the pattern is independent of the input. Such functions can thus be described by a directed acyclic graph (DAG) G, where each node v of the graph corresponds to some intermediate value ℓ_v that appears during the computation of the function, and the edges capture the computation: if ℓ_v is a function of previously computed values $\ell_{i_1}, \ldots, \ell_{i_\delta}$, then the nodes $i_1, \ldots, i_\delta$ are parents of v in G. For an iMHF F, we'll denote with $G(F)$ the underlying graph. For example $G(\mathsf{PBKDF2})$ is simply a path.

Graph Labeling Functions. Not only can an iMHF be captured by a graph as just outlined, we will actually construct iMHFs by first specifying a graph, and then defining a "labeling function" on top of it: Given a graph G with node set $V = [n] = \{1, 2, \ldots, n\}$, a hash function $H : \{0,1\}^* \to \{0,1\}^w$ and some input x, define the labeling of the nodes of G as follows: a source (a node v with indegree 0) has label $\ell_v(x) = H(v, x)$, a node v with parents $v_1 < v_2 < \cdots < v_\delta$ has label $\ell_v(x) = H(v, \ell_{v_1}(x), \ldots, \ell_{v_\delta}(x))$. For a DAG G with a unique sink s we define the labeling function of G as $f_G(x) = \ell_s(x)$. Note that using the convention from the previous paragraph, we have $G(f_G) = G$.

The Black Pebbling Game One of the main techniques for analyzing iMHF is to use pebbling games played on graphs. First introduced by Hewitt and Paterson [HP70] and Cook [Coo73] the (sequential) black pebbling game (and its relatives) have been used to great effect in theoretical computer science. Some early applications include space/time trade-offs for various computational tasks such as matrix multiplication [Tom78], the FFT [SS78,Tom78], integer multiplication [SS79b] and solving linear recursions [Cha73,SS79a]. More recently, pebbling games have been used for various cryptographic applications including proofs of space [DFKP15,RD16], proofs of work [DNW05,MMV13], leakage-resilient cryptography [DKW11a], garbled circuits [HJO$^+$16], one-time computable functions [DKW11b], adaptive security proofs [HJO$^+$16,JW16] and memory-hard functions [FLW13,AS15,AB16,AGK$^+$16]. It's also an active research topic in proof complexity (cf. the survey on `http://www.csc.kth.se/~jakobn/research/PebblingSurveyTMP.pdf`).

The black pebbling game is played over a fixed directed acyclic graph (DAG) $G = (V, E)$ in rounds. The goal of the game is to pebble all sink nodes of G (not necessarily simultaneously). Each round $i \geq 1$ is characterized by its pebbling configuration $P_i \subseteq V$ which denotes the set of currently pebbled nodes. Initially $P_0 = \emptyset$, i.e., all nodes are unpebbled. P_i is derived from the previous configuration P_{i-1} according to two simple rules. (1) A node v may be pebbled (added to P_i) if, in the previous configuration all of its parents were pebbled, i.e., $\mathsf{parents}(v) \subseteq P_{i-1}$. (2) A pebble can always be removed from P_i. In the sequential version rule (1) may be applied at most once per round while in the parallel version no such restriction applies. A sequence of configurations $P = (P_0, P_1, \ldots)$ is a (sequential) pebbling of G if it adheres to these rules and each sink node of G is contained in at least one configuration.

From a technical perspective, in this paper we investigate upper and lower bounds on various pebbling complexities of graphs, as they can be related to the cost of evaluating the "labeling function" f_G (to be defined below) in various computational models. In particular, let $\mathcal{P}_G$ and $\mathcal{P}_G^{\parallel}$ denote all valid sequential and parallel pebblings of G, respectively. We are interested in the *parallel* cumulative pebbling complexity of G, denoted $\Pi_{cc}^{\parallel}(G)$, and the sequential space-time complexity of G, denoted $\Pi_{st}(G)$, which are defined as

$$\Pi_{cc}^{\parallel}(G) = \min_{(P_1,\ldots,P_t)\in\mathcal{P}_G^{\parallel}} \sum_{i=1}^{t} |P_i| \qquad\qquad \Pi_{st}(G) = \min_{(P_1,\ldots,P_t)\in\mathcal{P}_G} t \cdot \max_i(|P_i|).$$

A main technical result of this paper is a family of graphs with high (in fact, as we'll discuss below, maximum possible) $\Pi_{cc}^{\parallel}$ complexity, and where the Π_{st}

complexity is not much higher than the $\Pi^{\parallel}_{cc}$ complexity.[3] Throughout, we'll denote with $\mathbb{G}_n$ the set of all DAGs on n nodes and with $\mathbb{G}_{n,d} \subseteq \mathbb{G}_n$ the DAGs where each node has indegree at most d.

Theorem 1. *There exists a family of DAGs $\{G_n \in \mathbb{G}_{n,2}\}_{n\in\mathbb{N}}$ where*

1. *the parallel cumulative pebbling complexity is*

$$\Pi^{\parallel}_{cc}(G_n) \in \Omega(n^2/\log(n))$$

2. *and where the sequential space-time complexity matches the parallel cumulative pebbling complexity up to a constant*

$$\Pi_{st}(G_n) \in O(n^2/\log(n)).$$

The lower bound on $\Pi^{\parallel}_{cc}$ in item *1.* above is basically optimal due to the following bound from [AB16]. [4]

Theorem 2 ([AB16, Thm. 8]). *For any constant $\epsilon > 0$ and sequence of DAGs $\{G_n \in \mathbb{G}_{n,\delta_n}\}_{n\in\mathbb{N}}$ it holds that*

$$\Pi^{\parallel}_{cc}(G_n) = o\left(\frac{\delta_n n^2}{\log^{1-\epsilon}}\right).$$

In particular if $\delta_n = O(\log^{1-\epsilon})$ then $\Pi^{\parallel}_{cc}(G_n) = o(n^2)$, and

$$\text{if } \delta_n = \Theta(1) \text{ then } \Pi^{\parallel}_{cc}(G_n) = o(n^2/\log^{1-\epsilon}(n)). \tag{1}$$

Pebbling vs. Memory-hardness. The reason to focus on the graph $G = G(F)$ underlying an iMHF F is that clean combinatorial properties of G – i.e., bounds on the pebbling complexities – imply both upper and lower bounds on the cost of evaluating F in various computational models. For upper bounds (i.e., attacks), no further assumption on F are required to make the transition from pebbling to computation cost. For lower bounds, we have to assume that there's no "shortcut" in computing F, and the only way is to follow the evaluation sequence as given by G. Given the current state of complexity theory, where not even superlinear lower bounds on evaluating any function in $\mathcal{NP}$ are known, we cannot hope to exclude such shortcuts unconditionally. Instead, we assume that the underlying hash function H is a random oracle and circuits are charged unit cost for queries to the random oracle.

[3] Note that $\Pi^{\parallel}_{cc}(G) \le \Pi_{st}(G)$ as parallelism can only help, and space-time complexity (i.e., number of rounds times the size of the largest state) is always higher than cumulative complexity (the sum of the sizes of all states).

[4] The statement below is obtained from the result in [AB16] by treating the core-memory ratio as a constant and observing that, trivially, at most n pebbles are on G during a balloon phase and at most n pebbles are placed in one step during a balloon phase.

For our lower bounds, we must insist on G having constant indegree. The reason is that in reality H must be instantiated with some cryptographic hash function like SHA1, and the indegree corresponds to the input length on which H is invoked. To evaluate H on long inputs, one would typically use some iterated construction like Merkle-Damgard, and the assumption that H behaves like a black-box that can only be queried once the entire input is known would be simply wrong in this case.

As advocated in [AS15], bounds on $\Pi^{\parallel}_{cc}(G)$ are a reasonable approximation for the cost of evaluating f_G in dedicated hardware, whereas a bound on $\Pi_{st}(G)$ gives an upper bound on the cost of evaluating f_G on a single processor machine. The reason [AS15] consider cumulative complexity for lower and space-time complexity for the upper bound is that when lower bounding the cost of evaluating f_G we do want to allow for amortization of the cost over arbitrary many instances,[5] whereas for our upper bound we don't want to make such an assumption. The reason we consider parallel complexity for the lower and only sequential for the upper bound is due to the fact that an adversary can put many (cheap) cores computing H on dedicated hardware, whereas for the upper bound we only want to consider a single processor machine.

If $\Pi^{\parallel}_{cc}(G)$ is sufficiently larger than $|V(G)|$ (in Theorem 1 it's almost quadratic), then the cost of evaluating f_G in dedicated hardware is dominated by the memory cost. As memory costs about the same on dedicated hardware and general purpose processors, if our G additionally satisfies $\Pi^{\parallel}_{cc}(G) \approx \Pi_{st}(C)$, then we get a function f_G whose evaluation on dedicated hardware is not much cheaper than evaluating it on an off the shelf machine (like a single core x86 architecture). This is exactly what the family from Theorem 1 achieves. We elaborate on these computational models and how they are related to pebbling in Appendix A.

On the positive side, previous to this work, the construction with the best asymptotic bounds was due to [AS15] and achieved $\Pi^{\parallel}_{cc}(G_n) \in \Omega(n^2/\log^{10}(n))$. However the exponent 10 (and the complexity of the construction) makes this construction uninteresting for practical purposes.

On the negative side [AB16,ACK$^+$16,AB17] have broken many popular iMHFs in a rather strong asymptotic sense. For example, in [AB16], the graph underlying Argon2i-A [BDK16], the winner of the recent Password Hashing Competition[6], was shown to have $\Pi^{\parallel}_{cc}$ complexity $\tilde{O}(n^{1.75})$. For Catena [FLW13] the upper bound $O(n^{5/3})$ is shown in [AB16]. In [AB17] these results were extended

---

[5] $\Pi^{\parallel}_{cc}$ satisfies a direct product property: pebbling k copies of G cost k times as much as pebbling G, i.e., $\Pi^{\parallel}_{cc}(G^k) = k \cdot \Pi^{\parallel}_{cc}(G)$, but this is not true for Π_{st} complexity.

[6] The Argon2 specification [BDK16] has undergone several revisions all of which are regularly referred to as "Argon2." To avoid confusion we follow [AB17] and use Argon2i-A [BDK15] to denote the version of Argon2i from the password hashing competition [PHC] and we use Argon2i-B [BDKJ16] to refer the version of Argon2 that is currently being considered for standardization by the Cryptography Form Research Group (CFRG) of the IRTF. We conjecture that the techniques introduced in this paper could also be used to establish tighter bounds for Argon2i-B. However, we leave this as an open challenge for future work.

to show that Argon2i-B [BDKJ16] has $\Pi_{cc}^{\parallel}(G) = O\left(n^{1.8}\right)$. Moreover [AB17] show that for random instances of these functions (which is how they are supposed to be used in practice) the attacks actually have far lower $\Pi_{cc}^{\parallel}$ than these asymptotic analyses indicate.

A New Generic Attack and Its Applications. In this work we improve on the attacks of [BK15,AS15,AB16] (Section 6). We give a new parallel pebbling strategy for pebbling DAGs which lack a generalization of depth-robustness. Next we investigate this property for the case of Argon2i-A, the three Balloon-Hashing variants and both Catena variants to obtain new upper bounds on their respective $\Pi_{cc}^{\parallel}$. For example, we further improve the upper bound on $\Pi_{cc}^{\parallel}$ for Argon2i-A and the Single Buffer variant of Balloon-Hashing from $\tilde{O}(n^{1.75})$ to $O(n^{1.708})$.

New Security Proofs. Complementing these results, in Section 5, we give the first security proofs for a variety of iMHFs. Hitherto the only MHF with a full security proof in a parallel computational model was [AS15] which employed relatively construction specific techniques. When restricted to sequential computation the results of [LT82,AS15] show that Catena has Π_{st} complexity $\Omega(n^2)$. Similar results are also shown for Argon2i-A and Balloon Hashing in [BCGS16].

In this work we introduce two new techniques for proving security of iMHFs. In the case of Argon2i-A and Argon2i-B we analyze its depth-robustness to show that its $\Pi_{cc}^{\parallel}$ is at least $\tilde{\Omega}(n^{5/3})$. The second technique involves a new combinatorial property called dispersion which we show to imply lower bounds on the $\Pi_{cc}^{\parallel}$ of a graph. We investigate the dispersion properties of the Catena and Balloon Hashing variants to show their $\Pi_{cc}^{\parallel}$ to be $\tilde{\Omega}(n^{1.5})$. Previously no (non-trivial) lower bounds on $\Pi_{cc}^{\parallel}$ were known for Argon2i-A, Catena or Balloon Hashing. Interestingly, our results show that Argon2i-A and Argon2i-B have better asymptotic security guarantees than Catena since $\Pi_{cc}^{\parallel} = \Omega(n^{5/3})$ for Argon2i-A and $\Pi_{cc}^{\parallel} = O(n^{13/8})$ for Catena.

While these lower bounds are significantly worse than what we might ideally hope for in a secure iMHF (e.g., $\Pi_{cc}^{\parallel} \geq \Omega(n^2/\log(n))$), we observe that, in light of our new attacks in Section 6, they are nearly tight. Unfortunately, together with the bounds on the sequential complexity of these algorithms our results do highlight a large asymptotic gap between the memory needed when computing the functions on parallel vs. sequential computational devices.

A table summarizing the asymptotic cumulative complexity of various iMHFs can be found in Table 1.

Depth-Robust Graphs. The results in this work rely on a new connection between the depth-robustness of a DAG and its $\Pi_{cc}^{\parallel}$ complexity. A DAG G is (e,d)-depth-robust if, after removing any subset of at most e nodes there remains a directed path of length at least d. First investigated by Erdös, Graham and Szemerédi [EGS75], several such graphs enjoying low indegree and increasingly extreme depth-robustness have been constructed in the past [EGS75,PR80,Sch82,Sch83,MMV13] mainly in the context of proving lower-bounds on circuit complexity and Turing machine time. Depth-robustness has been used as a key tool in the construction

Algorithm	Lowerbound	Upperbound	Appearing In
Argon2i-A		$\tilde{O}\left(n^{1.75}\right)$	[AB16]
Argon2i-A	$\tilde{\Omega}\left(n^{1.\bar{6}}\right)$	$\tilde{O}\left(n^{1.708}\right)$	This Work
Argon2i-B		$O\left(n^{1.8}\right)$	[AB17]
Argon2i-B	$\tilde{\Omega}\left(n^{1.\bar{6}}\right)$		This Work
Balloon-Hashing: Linear and Double Buffer (DB)		$O\left(n^{1.67}\right)$	[AB16]
Balloon-Hashing: Linear and Double Buffer(DB)	$\tilde{\Omega}\left(n^{1.5}\right)$	$\tilde{O}\left(n^{1.625}\right)$	This Work
Balloon-Hashing: Single Buffer (SB)		$\tilde{O}\left(n^{1.75}\right)$	[AB16]
Balloon-Hashing: Single Buffer (SB)	$\tilde{\Omega}\left(n^{1.\bar{6}}\right)$	$\tilde{O}\left(n^{1.708}\right)$	This Work
Catena: Dragonfly		$O\left(n^{1.67}\right)$	[AB16]
Catena: Dragonfly	$\tilde{\Omega}\left(n^{1.5}\right)$	$\tilde{O}\left(n^{1.625}\right)$	This Work
Catena: Butterfly		$O\left(n^{1.67}\right)$	[AB16]
Catena: Butterfly	$\tilde{\Omega}\left(n^{1.5}\right)$	$o\left(n^{1.625}\right)$	This Work
[AS15]	$\Omega\left(\frac{n^2}{\log^{10} n}\right)$		[AS15]
Theorem 1	$\Omega\left(\frac{n^2}{\log n}\right)$		This Work
Arbitrary iMHF		$O\left(\frac{n^2 \log\log n}{\log n}\right)$	[AB16]

Table 1: Overview of the asymptotic cumulative complexity of various iMHF.

of cryptographic objects like proofs of sequential work [MMV13]. In fact depth-robust graphs were already used as a building block in the construction of a high $\Pi_{cc}^{\parallel}$ graph in [AS15].

Depth-Robustness and $\Pi_{cc}^{\parallel}$. While the flavour of the results in this work are related to those of [AS15] the techniques are rather different. As mentioned above already, they stem from a new tight connection between depth-robustness and $\Pi_{cc}^{\parallel}$. A special case of this connection shows that if G is (e,d)-depth-robust, then its $\Pi_{cc}^{\parallel}$ can be lower bounded as

$$\Pi_{cc}^{\parallel}(G) \geq e \cdot d \ .$$

This complements a result from [AB16], who gives a pebbling strategy which is efficient for graphs of low depth-robustness (we give the exact statement in Theorem 10 below), thus a DAG has high $\Pi_{cc}^{\parallel}$ if and only if it is very depth-robust.

Moreover, we give a new tool for reducing the indegree of a DAG while not reducing the $\Pi_{cc}^{\parallel}$ of the resulting graph (in terms of its size). Together these results directly have some interesting consequences

– The family of DAGs $\{G_n \in \mathbb{G}_{n,\log(n)}\}_{n\in\mathbb{N}}$ from Erdös et al. [EGS75] have optimally high $\Pi_{cc}^{\parallel}(G_n) \in \Omega(n^2)$.

– Using our indegree reduction we can turn the above family of $\log(n)$ indegree into a family of indegree 2 DAGs $\{G'_n \in \mathbb{G}_{(n,2)}\}_{n\in\mathbb{N}}$ with $\Pi^{\parallel}_{cc}(G'_n) \in \Omega(n^2/\log(n))$, which by Theorem 2 is optimal for constant indegree graphs.

Data-Dependent MHFs. One can naturally extend the $\Pi^{\parallel}_{cc}$ notion also to "dynamic" graphs – where some edges are only revealed as some nodes are pebbled – in order to analyse data-dependent MHFs (dMHF) like `scrypt`. In this model, [ACK+16] show that $\Pi^{\parallel}_{cc}(\texttt{scrypt}) = \Omega(n^2/\log^2(n))$. Unfortunately unlike for iMHFs, for dMHFs we do not have a proof that a lower bound on $\Pi^{\parallel}_{cc}$ implies roughly the same lower bound on the cumulative memory complexity in the random oracle model.[7] Recently a "direct" proof (i.e., avoiding pebbling arguments) – showing that `scrypt` has optimal cumulative memory complexity $\Omega(n^2)$ – has been announced, note that this bound is better than what we can hope to achieve for iMHFs (as stated in Theorem 2). Unfortunately, the techniques that have now been developed to analyse dMHFs seem not to be useful for the iMHF setting.

2 Pebbling Complexities and Depth-Robustness of Graphs

We begin by fixing some common notation. We use the sets $\mathbb{N} = \{0, 1, 2, \ldots\}$, $\mathbb{N}^+ = \{1, 2, \ldots\}$, and $\mathbb{N}_{\geq c} = \{c, c+1, c+2, \ldots\}$ for $c \in \mathbb{N}$. Further, we also use the sets $[c] := \{1, 2, \ldots, c\}$ and $[b, c] = \{b, b+1, \ldots, c\}$ where $b \in \mathbb{N}$ with $b \leq c$. For a set of sets $A = \{B_1, B_2, \ldots, B_z\}$ we use the notation $||A|| := \sum_i |B_i|$.

2.1 Depth-Robust Graphs

We say that a directed acyclic graph (DAG) $G = (V, E)$ has *size* n if $|V| = n$. A node $v \in V$ has indegree $\delta = \mathsf{indeg}(v)$ if there exist δ incoming edges $\delta = |(V \times \{v\}) \cap E|$. More generally, we say that G has indegree $\delta = \mathsf{indeg}(G)$ if the maximum indegree of any node of G is δ. A node with indegree 0 is called a source node and one with no outgoing edges is called a sink. We use $\mathsf{parents}_G(v) = \{u \in V : (u, v) \in E\}$ to denote the parents of a node $v \in V$. In general, we use $\mathsf{ancestors}_G(v) = \bigcup_{i\geq 1} \mathsf{parents}^i_G(v)$ to denote the set of all ancestors of v — here, $\mathsf{parents}^2_G(v) = \mathsf{parents}_G(\mathsf{parents}_G(v))$ denotes the grandparents of v and $\mathsf{parents}^{i+1}_G(v) = \mathsf{parents}_G\left(\mathsf{parents}^i_G(v)\right)$. When G is clear from context we will simply write $\mathsf{parents}$ ($\mathsf{ancestors}$). We denote the set of all sinks of G with $\mathsf{sinks}(G) = \{v \in V : \nexists (v, u) \in E\}$ — note that $\mathsf{ancestors}(\mathsf{sinks}(G)) = V$. We often consider the set of all DAGs of equal size $\mathbb{G}_n = \{G = (V, E) : |V| = n\}$ and often will bound the maximum indegree $\mathbb{G}_{n,\delta} = \{G \in \mathbb{G}_n : \mathsf{indeg}(G) \leq \delta\}$.

[7] [ACK+16] introduces a combinatorial conjecture, which if true, means that lower bounds on $\Pi^{\parallel}_{cc}$ translate to cumulative memory complexity. At this point a strong variant of the conjecture has already been refuted, the state of the conjecture is updated in the eprint version [ACK+16] of the paper.

For directed path $p = (v_1, v_2, \ldots, v_z)$ in G its length is the number of nodes it traverses $\mathsf{length}(p) := z$. The depth $d = \mathsf{depth}(G)$ of DAG G is the length of the longest directed path in G.

We will often consider graphs obtained from other graphs by removing subsets of nodes. Therefore if $S \subset V$ then we denote by $G - S$ the DAG obtained from G by removing nodes S and incident edges. The following is a central definition to our work.

Definition 1 (Depth-Robustness). *For $n \in \mathbb{N}$ and $e, d \in [n]$ a DAG $G = (V, E)$ is (e, d)-depth-robust if*

$$\forall S \subset V \quad |S| \leq e \Rightarrow \mathsf{depth}(G - S) \geq d.$$

We will make use of the following lemma due to Erdös, Graham and Szemerédi [EGS75], who showed how to construct a family of log indegree DAGs with extreme depth-robustness.

Theorem 3 ([EGS75]). *For some fixed constants $c_1, c_2, c_3 > 0$ there exists an infinite family of DAGs $\{G_n \in \mathbb{G}_{n, c_3 \log(n)}\}_{n=1}^{\infty}$ such that G_n is $(c_1 n, c_2 n)$-depth-robust.*

2.2 Graph Pebbling

We fix our notation for the parallel graph pebbling game following [AS15].

Definition 2 (Parallel/Sequential Graph Pebbling). *Let $G = (V, E)$ be a DAG and let $T \subseteq V$ be a target set of nodes to be pebbled. A* pebbling configuration *(of G) is a subset $P_i \subseteq V$. A legal* parallel *pebbling of T is a sequence $P = (P_0, \ldots, P_t)$ of pebbling configurations of G where $P_0 = \emptyset$ and which satisfies conditions 1 & 2 below. A* sequential *pebbling additionally must satisfy condition 3.*

1. *At some step every target node is pebbled (though not necessarily simultaneously).*
$$\forall x \in T \; \exists z \leq t \quad : \quad x \in P_z.$$
2. *Pebbles are added only when their predecessors already have a pebble at the end of the previous step.*
$$\forall i \in [t] \quad : \quad x \in (P_i \setminus P_{i-1}) \; \Rightarrow \; \mathsf{parents}(x) \subseteq P_{i-1}.$$
3. *At most one pebble placed per step.*
$$\forall i \in [t] \quad : \quad |P_i \setminus P_{i-1}| \leq 1 \ .$$

We denote with $\mathcal{P}_{G,T}$ and $\mathcal{P}^{\parallel}_{G,T}$ the set of all legal sequential and parallel pebblings of G with target set T, respectively. Note that $\mathcal{P}_{G,T} \subseteq \mathcal{P}^{\parallel}_{G,T}$. We will be mostly interested in the case where $T = \mathsf{sinks}(G)$ and then will simply write $\mathcal{P}_G$ and $\mathcal{P}^{\parallel}_G$.

Definition 3 (Time/Space/Cumulative Pebbling Complexity). *The time, space, space-time and cumulative complexity of a pebbling* $P = \{P_0, \ldots, P_t\} \in \mathcal{P}_G^{\parallel}$ *are defined to be:*

$$\Pi_t(P) = t \quad \Pi_s(P) = \max_{i \in [t]} |P_i| \quad \Pi_{st}(P) = \Pi_t(P) \cdot \Pi_s(P) \quad \Pi_{cc}(P) = \sum_{i \in [t]} |P_i| \ .$$

For $\alpha \in \{s, t, st, cc\}$ *and a target set* $T \subseteq V$*, the sequential and parallel pebbling complexities of* G *are defined as*

$$\Pi_\alpha(G, T) = \min_{P \in \mathcal{P}_{G,T}} \Pi_\alpha(P) \qquad \text{and} \qquad \Pi_\alpha^{\parallel}(G, T) = \min_{P \in \mathcal{P}_{G,T}^{\parallel}} \Pi_\alpha(P) \ .$$

When $T = \mathsf{sinks}(G)$ *we simplify notation and write* $\Pi_\alpha(G)$ *and* $\Pi_\alpha^{\parallel}(G)$*.*

It follows from the definition that for $\alpha \in \{s, t, st, cc\}$ and any G the parallel pebbling complexity is always at most as high as the sequential, i.e., $\Pi_\alpha(G) \geq \Pi_\alpha^{\parallel}(G)$, and cumulative complexity is at most as high as space-time complexity, i.e., $\Pi_{st}(G) \geq \Pi_{cc}(G)$ and $\Pi_{st}^{\parallel}(G) \geq \Pi_{cc}^{\parallel}(G)$.

In this work we will consider constant in-degree DAGs $\{G_n \in \mathbb{G}_{n,\Theta(1)}\}_{n \in \mathbb{N}}$, and will be interested in the complexities $\Pi_{st}(G_n)$ and $\Pi_{cc}^{\parallel}(G_n)$ as these will capture the cost of evaluating the labelling function derived from G_n on a single processor machine (e.g. a x86 processor on password server) and amortized AT complexity (which is a good measure for the cost of evaluating the function on dedicated hardware), respectively.

Before we state our main theorem let us observe some simple facts. Every n-node graph can be pebbled in n steps, and we cannot have more than n pebbles on an n node graph, thus

$$\forall G_n \in \mathbb{G}_n \ : \ \Pi_{cc}^{\parallel}(G_n) \ \leq \ \Pi_{st}(G_n) \ \leq \ n^2.$$

This upper bound is basically matched for the complete graph $K_n = (V = [n], E = \{(i,j) \ : 1 \leq i < j \leq n\})$ as

$$n(n-1)/2 \ \leq \ \Pi_{cc}^{\parallel}(K_n) \ \leq \ \Pi_{st}(K_n) \ \leq \ n^2.$$

Graph K_n has the desirable properties that its Π_{st} is within a constant factor to its $\Pi_{cc}^{\parallel}$ complexity and that its $\Pi_{cc}^{\parallel}$ complexity is maximally high. Unfortunately, K_n has very high indegree, which makes it useless for our purpose to construct memory-hard functions. The path $Q_n = (V = [n], E = \{(i, i+1) : 1 \leq i \leq n-1\})$ on the other hand has indegree 1 and its Π_{st} is even exactly as large as its $\Pi_{cc}^{\parallel}$ complexity. Unfortunately it has very low pebbling complexity

$$\Pi_{cc}^{\parallel}(Q_n) \ = \ \Pi_{st}(Q_n) \ = \ n$$

which means that in the labelling function we get from Q_n (which is basically PBKDF2 discussed in the introduction) the evaluation cost will not be dominated by the memory cost even for large n. As stated in Theorem 1, in this paper we construct a family of graphs $\{G_n \in \mathbb{G}_{n,2}\}_{n\in\mathbb{N}}$ which satisfies all three properties at once: (1) the graphs have indegree 2 (2) the parallel cumulative pebbling complexity is $\Pi^{\parallel}_{cc}(G_n) \in \Omega(n^2/\log(n))$, which by by Theorem 2 is optimal for constant indegree graphs, and (3) $\Pi_{st}(G_n)$ is within a constant factor of $\Pi^{\parallel}_{cc}(G_n)$.

3 Depth-Robustness Implies High $\Pi^{\parallel}_{cc}$

In this section we state and prove a theorem which lowerbounds the $\Pi^{\parallel}_{cc}$ of a given DAG G in terms of its depth robustness.

Theorem 4. *Let G be an (e,d)-depth-robust DAG, then $\Pi^{\parallel}_{cc}(G) > ed$.*

Proof. Let $(P_1,\ldots,P_m)$ be a parallel pebbling of minimum complexity, i.e., $\sum_{i=1}^{m}|P_i| = \Pi^{\parallel}_{cc}(G)$. For any d, we'll show that there exists a set B of size $|B| \le \Pi^{\parallel}_{cc}(G)/d$ such that there's no path of length d in $G - B$, or equivalently, G is not $(\Pi^{\parallel}_{cc}(G)/d, d)$-depth-robust, note that this implies the theorem.

For $i \in [d]$ define $B_i = P_i \cup P_{i+d} \cup P_{i+2d} \ldots$. We observe that by construction $\sum_{i=0}^{d-1}|B_i| \le \sum_{i=1}^{m}|P_i| = \Pi^{\parallel}_{cc}(G)$, so the size of the B_i's is $\le \Pi^{\parallel}_{cc}(G)/d$ on average, and the smallest B_i has size at most this. Let B be the smallest B_i, as just outlined $|B| \le \Pi^{\parallel}_{cc}(G)/d$.

It remains to show that $G - B$ has no path of length d. For this consider any path $v_1,\ldots,v_d$ of length d in G. Let j be minimal such that $v_d \in P_j$ (so v_d is pebbled for the first time in round j of the pebbling). It then must be the case that $v_{d-1} \in P_{j-1}$ (as to pebble v_d in round j there must have been a pebble on v_{d-1} in round $j-1$). In round $j-2$ either the pebble on v_{d-1} was already there, or there was a pebble on v_{d-2}. This argument shows that each of the pebbling configurations $\{P_{j-d+1},\ldots,P_j\}$ must contain at least one node from $v_1,\ldots,v_d$. As B contains each dth pebbling configuration, B contains at least one of these pebbling configurations $\{P_{j-d+1},\ldots,P_j\}$. Specifically we can find $j-d+1 \le k \le j$ s.t $P_k \subseteq B$, thus the path $v_1,\ldots,v_d$ is not contained entirely in $G - B$. □

An immediate implication of Theorem 4 and Theorem 3 is that there is an infinite family of DAGs with maximal $\Pi^{\parallel}_{cc}(G) = \Omega(n^2)$ whose indegree scales with $\log n$. Note that this means that allowing indegree as small as $O(\log(n))$ is sufficient to get DAGs whose $\Pi^{\parallel}_{cc}$ is within a constant factor of the n^2 upper bound on $\Pi^{\parallel}_{cc}$ for any n node DAG. In the next section we will show how to reduce the indegree to $O(1)$ while only reducing $\Pi^{\parallel}_{cc}(G)$ by a factor of $O(\log(n))$.

Corollary 1 (of Theorem 4 and Theorem 3). *For some constants $c_1, c_2 > 0$ there exists an infinite family of DAGs $\{G_{n,\delta} \in \mathbb{G}_{n,\delta}\}_{n=1}^{\infty}$ with $\delta \le c_1\log(n)$ and $\Pi^{\parallel}_{cc}(G) \ge c_2 n^2$. This is optimal in the sense that for any family $\{\delta_n \in [n]\}_{n=1}^{\infty}$ and $\{J_n \in \mathbb{G}_{n,\delta_n}\}_{n=1}^{\infty}$ it holds that $\Pi^{\parallel}_{cc}(J_n) \in O(n^2)$. Moreover if $\delta_n = o(\log(n)/\log\log(n))$ then $\Pi^{\parallel}_{cc}(J_n) = o(n^2) = o(\Pi^{\parallel}_{cc}(G_n))$.*

Corollary 2 lower bounds the cost of pebbling a target set T given a starting pebbling configuration S. In particular, if the ancestors of T in $G-S$ induce an (e,d)-depth-robust DAG then the pebbling cost is at least $\Pi^{\parallel}_{cc}(G-S,T) \geq ed$. We will use Corollary 2 to lower bound $\Pi^{\parallel}_{cc}$ for iMHFs like Argon2i and SB.

Corollary 2 (of Theorem 4). *Given a DAG $G=(V,E)$ and subsets $S,T \subset V$ such that $S \cap T = \emptyset$ let $G' = G - (V \setminus \mathsf{ancestors}_{G-S}(T))$. If G' is (e,d)-depth robust then the cost of pebbling $G-S$ with target set T is $\Pi^{\parallel}_{cc}(G-S,T) > ed$.*

Proof. Note that $\Pi^{\parallel}_{cc}(G-S,T) \geq \Pi^{\parallel}_{cc}(G')$ since we will need to pebble every node in the set $\mathsf{ancestors}_{G-S}(T) = V(G')$ to reach the target set T in $G-S$. By Theorem 4, we have $\Pi^{\parallel}_{cc}(G') > ed$. □

Corollary 3 states that it remains expensive to pebble any large enough set of remaining nodes in a depth-robust graph even if we are permitted to first remove an arbitrary node set of limited size. An application of Corollary 3 might involve analysing the cost of pebbling stacks of depth-robust graphs. For example if there are not enough pebbles on the graph at some point in time then there must be some layers with few pebbles. If we can then show that many of the nodes on those layers will eventually need to be (re)pebbled then we can use this lemma to show that the remaining pebbling cost incurred by these layers is large.

Corollary 3 (of Theorem 4). *Let DAG $G=(V,E)$ be (e,d)-depth-robust and let $S,T \subset V$ such that*

$$|S| \leq e \quad \text{and} \quad T \cap S = \emptyset.$$

Then the cost of pebbling $G-S$ with target set T is $\Pi^{\parallel}_{cc}(G-S,T) > (e-|S|)\,(d-|\mathsf{ancestors}_{G-S}(T)|)$.

Proof. Let $G' = G - (V - ancestors_{G-S}(T))$ and observe that G' is, at minimum, $(e-|S|, d-|\mathsf{ancestors}_{G-S}(T)|)$-depth robust. By Corollary 2 we have $\Pi^{\parallel}_{cc}(G-S,T) \geq \Pi^{\parallel}_{cc}(G') > (e-|S|)\,(d-|\mathsf{ancestors}_{G-S}(T)|)$. □

We remark that Theorem 4 is a special case of Corollary 3 by setting $S=\emptyset$ letting $T = \mathsf{sinks}(G)$. Recall that $\Pi^{\parallel}_{cc}(G)$ is the parallel pebbling of minimal cumulative cost when pebbling all sinks of G, this requires pebbling all nodes of G at least once.

4 Indegree Reduction: Constant Indegree with Maximal $\Pi^{\parallel}_{cc}$

In this section we use the result from the previous section to show a new, more efficient, degree-reduction lemma. We remark that Lemma 1 is similar to [AS15, Lemma 9] in that both reductions replace high indegree nodes v in G with a path. However, we stress two key differences between the two results. First, our focus is on reducing the indegree while preserving depth-robustness. By contrast, [AS15, Lemma 9] focuses directly on preserving $\Pi^{\parallel}_{cc}$. Second, we note that the guarantee of [AS15, Lemma 9] is weaker in that it yields a reduced indegree graph G' whose size grows by a factor of indeg ($n' \leq n \times \mathsf{indeg}$) while $\Pi^{\parallel}_{cc}$ can drop by a factor of indeg — [AS15, Lemma 9] shows that $\Pi^{\parallel}_{cc}(G') \geq \frac{\Pi^{\parallel}_{cc}(G)}{\mathsf{indeg}-1}$. By contrast, setting $\gamma = \mathsf{indeg}$ in Lemma 1 yields a reduced indegree graph G' whose size grows by a factor of $2 \times \mathsf{indeg}$ ($n' \leq 2n \times \mathsf{indeg}$) and better depth-robustness $(e', d') = (e, d \times \mathsf{indeg})$. In particular, when we apply Theorem 4 the lower-bound $\Pi^{\parallel}_{cc}(G') \geq ed \times \mathsf{indeg}$ *improves* by a factor of indeg when compared with the original graph G.

Lemma 1. *Let G be a (e,d)-depth-robust DAG. For $\gamma \in \mathbb{Z}_{\geq 0}$ there exists a $(e, d\gamma)$-depth-robust DAG G' with*

$$\mathsf{size}(G') \leq (\mathsf{indeg}(G)+\gamma)\cdot\mathsf{size}(G) \ , \qquad \mathsf{indeg}(G') = 2 \ \textit{and} \qquad \Pi_{st}(G') \leq \frac{\mathsf{size}(G')^2}{\gamma} .$$

Proof. Fix a $\gamma \in \mathbb{Z}_{\geq 0}$ and let $\delta = \mathsf{indeg}(G)$. We identify each node in V' with an element of the set $V \times [\delta + \gamma]$ and we write $\langle v, j \rangle \in V'$. For every node $v \in V$ with $\alpha_v := \mathsf{indeg}(v) \in [0, \delta]$ we add the path $p_v = (\langle v, 1\rangle, \langle v, 2\rangle, \ldots, \langle v, \alpha_v + \gamma\rangle)$ of length $\alpha_v + \gamma$. We call v the *genesis node* and p_v its *metanode*. In particular $V' = \cup_{v \in V} p_v$. Thus G has size at most $(\delta + \gamma)n$.

Next we add the remaining edges. Intuitively, for the i^{th} incoming edge (u, v) of v we add an edge to G' connecting the end of the metanode of u to the i^{th} node in the metanode of v. More precisely, for every $v \in V$, $i \in [\mathsf{indeg}(v)]$ and edge $(u_i, v) \in E$ we add edge $(\langle u_i, \mathsf{indeg}(u_i) + \gamma\rangle, \langle v, i\rangle)$ to E'. It follows immediately that G' has indegree (at most) 2.

Fix any node set $S \subset V'$ of size $|S| \leq e$. Then at most e metanodes can share a node with S. For each such metanode remove its genesis node in G. As G is (e, d)-depth-robust we are still left with a path p of length (at least) d in G. But that means that after removing S from G' there must remain a path p' in G' running through all the metanodes of p and $|p'| \geq |p|\gamma \geq d\gamma$. In other words G' is $(e, d\gamma)$-depth-robust.

To see that $\Pi_{st}(G') \leq \mathsf{size}(G')^2/\gamma$ we simply pebble G' in topological order. We note that we never need to keep more than one pebble on any metanode $p_v = (\langle v, 1\rangle, \langle v, 2\rangle, \ldots, \langle v, \alpha_v + \gamma\rangle)$ with $\alpha_v = \mathsf{indeg}(v)$. Once we pebble the last node $\langle v, \alpha_v + \gamma\rangle$ we can permanently discard any pebbles on the rest of p_v since $\langle v, \alpha_v + \gamma\rangle$ is the only node with outgoing edges. □

Proof of Theorem 1. Theorem 1 follows by applying Lemma 1 to the family from Theorem 3 with $\gamma = \mathsf{indeg} = \log n$. We get that for some fixed constants $c_1, c_2 > 0$ there exists an infinite family of indegree 2 DAGs $\{G_n \in \mathbb{G}_{n,2}\}_{n=1}^{\infty}$ where G_n is $(c_1 n/\log n, c_2 n)$-depth robust and $\Pi_{st}(G_n) \leq O\big(n^2/\log(n)\big)$. By Theorem 4 then $\Pi^{\parallel}_{cc}(G_n) > (c_1 c_2) n^2/\log(n)$, which is basically optimal for constant indegree DAGs by Theorem 2. □

5 Security Proofs of Candidate iMHFs

On the surface, in this and the next section we give both security proofs and nearly optimal attacks for several of the most prominent iMHF proposals. That is we show both lower and (relatively tight) upperbounds on their asymptotic memory-hardness in the PROM. However, more conceptually, we also introduce two new proof techniques for analysing the depth-robustness of DAGs as well as a new very memory-efficient class of algorithms for pebbling a DAG improving on the techniques used in [AB16]. Indeed for all candidates considered the attack in the next section is almost optimal in light of the accompanying security proofs in this section.

More specifically, in the first subsection we prove bounds for a class of random graphs which generalize the Argon2i-A construction [BDK16] and the Single Buffer (SB) variant of Balloon Hashing [BCGS16]. To prove the lowerbound we use a simple and clean new technique for bounding the depth-robustness of a random DAG. In particular, we show that a random DAG is almost certainly $\left(e, \tilde{\Omega}\left(n^2/e^2\right)\right)$-depth robust for any $e > \sqrt{n}$. Combined with Theorem 4 we could immediately obtain a lower bound of $\tilde{\Omega}\left(n^{1.5}\right)$. We can improve the lower bound to $\tilde{\Omega}\left(n^{5/3}\right)$ by introducing a stronger notion of depth-robustness that we call block depth-robustness.

In the second subsection we prove bounds for a family of layered graphs which generalize both of the Catena constructions [FLW13] as well as Linear (Lin) and Double Buffer (DB) variants of Balloon Hashing [BCGS16]. In particular, we introduce a new technique for proving lowerbounds on the cumulative pebbling complexity of a graph without going through the notion of depth-robustness. For example the (single layer) version of the Catena Dragonfly graph has the worst possible depth-robustness of any graph of linear depth. This shows that (in the lower but still non-trivial regimes of) cumulative complexity alternative combinatorial structures exist besides depth-robustness that can also confer some degree of pebbling complexity.

5.1 Lowerbounding the CC of Random DAGs.

We begin by defining a (n, δ, w)-random DAG, the underlying DAGs upon which Argon2i-A and SB are based. The memory window parameter w specifies the intended memory usage and throughput of the iMHF — the cost of the naïve pebbling algorithm is $\Pi^{\parallel}_{cc}(\mathcal{N}) = wn$. In particular, a t-pass Argon2i-A iMHF is based on a $(n, 2, n/t)$-random DAG. Similarly, a t-pass Single-Buffer (SB) iMHF [BCGS16] is based on a $(n, 20, n/t)$-random DAG. In this section we focus on the $t = 1$-pass variants of the Argon2i-A and [BCGS16] iMHFs.

Definition 4 ((n, δ, w)-random DAG). *Let $n \in \mathbb{N}$, $1 < \delta < n$, and $1 \leq w \leq n$ such that w divides n. An (n, δ, w)*-random DAG *is a randomly generated directed acyclic (multi)graph with n nodes $v_1, \ldots, v_n$ (which we identify with the set $[n]$ according to there topological order) and with maximum in-degree δ for each node. The graph has directed edges (v_i, v_{i+1}) for $1 \leq i < n$ and random forward* *edges* $(v_{r(i,1)}, v_i), \ldots, (v_{r(i,\delta-1)}, v_i)$ *for each node* v_i. *Here,* $r(i,j)$ *is independently chosen uniformly at random from the set* $[\max\{0, i-w\}, i-1]$.

Theorem 5 states that for a (n, δ, n)-random DAG G such as Argon2i-A or SB we almost certainly have $\Pi_{cc}^{\parallel}(G) = \tilde{\Omega}\left(n^{5/3}\right)$.

Theorem 5. *Let* G *be a* (n, δ, n)*-random DAG then, except with probability* $o(n^{-7})$, *we have*

$$\Pi_{cc}^{\parallel}(G) = \tilde{\Omega}\left(n^{5/3}\right) .$$

Security Lower Bound. To prove the lower bound we rely on a slightly stricter notion of depth robustness. Given a node v let $N(v,b) = \{v-b+1, \ldots, v\}$ denote a segment of b consecutive nodes ending at v and given a set $S \subseteq V(G)$ let $N(S,b) = \bigcup_{v \in S} N(v,b)$. We say that a DAG G is (e,d,b)-block depth-robust if for every set $S \subseteq V(G)$ of size $|S| \leq e$ we have $\mathsf{depth}(G - N(S,b)) \geq d$. Notice that when $b = 1$ (e,d,b)-block-depth robustness is equivalent to (e,d)-depth-robustness. However, when $b > 1$ (e,d,b)-block-depth robustness is a strictly stronger notion since the set $N(S,b)$ may have size as large as $|N(S,b)| = eb$. [8]

The proof of Theorem 5 relies on Lemma 2, which states that for any $e \geq \sqrt{n}$, with high probability, a $(n,2,n)$-random DAG G will be (e,d,b)-block depth-robust with $d = \frac{n^2}{e^2 \mathsf{polylog}(n)}$ and $b = n/(20e)$. By contrast Lemma 9 states that G will be (e,d)-reducible with $d = \tilde{O}\left(n^2/e^2\right)$.

Lemma 2. *For any* $e \geq \sqrt{n}$ *any any* $\delta \geq 2$ *a* (n, δ, n)*-random DAG will be* $\left(e, \Omega\left(\frac{n^2}{e^2 \log(n)}\right), \frac{n}{20e}\right)$*-block depth robust except with negligible probability in* n.

Setting $e = \sqrt{n}$ in Lemma 2 and applying Theorem 4 already implies that $\Pi_{cc}^{\parallel}(G) = \tilde{\Omega}\left(n^{1.5}\right)$. To obtain the stronger bound in Theorem 5 we rely on Corollary 2 combined with a more sophisticated argument exploiting block depth-robustness.

In more detail, let G be an (n, δ, n)-random DAG and let t_j denote the first time we place a pebble on node j. Observe that, since G contains all edges of the form $(j, j+1)$ it must be that $t_{j+i} - t_j \geq i$ in any legal pebbling of G. We will show that for any $j > n/2$ a legal pebbling must (almost certainly) incur a cost of $\tilde{\Omega}\left(n^{4/3}\right)$ between pebbling steps t_j and t_{j+2k} where $k = \tilde{\Theta}\left(n^{2/3}\right)$. That is $\sum_{t=t_j}^{t_{j+2k}} |P_t| = \tilde{\Omega}\left(n^{4/3}\right)$ for any legal pebbling of G. Thus, $\sum_{t=t_{n/2+1}}^{t_n} |P_t| = \tilde{\Omega}\left(n^{4/3}\frac{n/2}{k}\right) = \tilde{\Omega}\left(n^{5/3}\right)$. In the remaining discussion we set $e = \tilde{\Omega}\left(n^{2/3}\right)$, $d = \tilde{\Omega}\left(n^{2/3}\right)$ and $b = \tilde{\Omega}\left(n^{1/3}\right)$.

To show that $\sum_{t=t_j}^{t_{j+2k}} |P_t| = \tilde{\Omega}\left(n^{4/3}\right)$ we consider two cases: we either have $|P_t| \geq e/2 = \tilde{\Omega}\left(n^{2/3}\right)$ pebbles on the DAG during each round $t_j \leq t \leq t_{j+k}$, or we do not. In the first case we trivially have $\sum_{t=t_j}^{t_{j+2k}} |P_t| \geq ke/2 = \tilde{\Omega}\left(n^{4/3}\right)$.

[8] In particular, $(e, d, b \geq 1)$-block depth robustness implies (e,d)-depth robustness. However, (e,d)-depth robustness only implies $(e/b, d, b)$-block depth robustness.

The second case is the trickier one to handle. To address it we essentially show that if, at some moment t', few pebbles are left on G then between t' and t_{j+2k} it must be that (in particular) a depth-robust sub-graph of G was pebbled which we know requires a high pebbling cost. In more detail, suppose at some moment $t' \in [t_j, t_{j+k}]$ only $|P_t| < e/2$ pebbles remain on G. Then we consider the sub-graph H induced by the node set $\mathsf{ancestors}_{G_1 - N(P_{t'}, b)}([j+k+1, j+2k])$. We observe that, on the one hand, H must be fully pebbled during the interval $[t', t_{j+2k}]$. On the other hand, we observe that $G_1 = G - \{n/2+1, \ldots, n\}$ is a $(n/2, \delta, n/2)$-random DAG and, hence, by Lemma 2, G_1 is (almost certainly) (e, d, b)-block depth robust with $e = \tilde{\Omega}\left(n^{2/3}\right)$, $d = \Omega\left(\frac{n^{2/3}}{\log(n)}\right)$ and $b = \tilde{\Omega}\left(n^{1/3}\right)$. By exploiting the block depth robustness of G_1 we can show that H must itself be $\left(\tilde{\Omega}\left(n^{2/3}\right), \tilde{\Omega}\left(n^{2/3}\right)\right)$-depth robust. But then by Corollary 2 we get that H has cumulative complexity $\tilde{\Omega}(n^{4/3})$ and so have

$$\sum_{t=t_{j+k+1}}^{t_{j+2k}} |P_t| \geq \Pi_{cc}^{\parallel}\left(G_1 - P_{t'}, [j+k+1, j+2k]\right) \geq \tilde{\Omega}\left(n^{4/3}\right) .$$

The proofs of Lemma 2 and Theorem 5 are in Appendix B. We now make a couple of observations about Lemma 2 and Theorem 5.

1. The lower bounds from Lemma 2 and Theorem 5 also apply to Argon2i-B. An Argon2i-B DAG G is similar to an (n, δ, n)-random DAG except that the randomly chosen forward edge $(r(i), i)$ for each node i is not chosen from the uniform distribution. However, these edges are still chosen independently and for each pair $j < i$ we still have $\Pr[r(i) = j] = \Omega(1/i)$. These are the only properties we used in the proofs of Lemma 2 and Theorem 5. Thus, essentially the same analysis shows that (whp) an Argon2i-B DAG G is $\left(e, \Omega\left(n^2/e^2\right), \frac{n}{20e}\right)$-block depth robust and that $\Pi_{cc}^{\parallel}(G) = \tilde{\Omega}\left(n^{5/3}\right)$.
2. The lower bound from Lemma 2 is tight up to polylogarithmic factors. In particular, a generalization of an argument of Alwen and Blocki [AB16] shows that a (n, δ, n)-random DAG is $\left(e, \tilde{\Omega}\left(\frac{n^2}{e^2}\right)\right)$-reducible — see Lemma 9. However, this particular upper bound does not extend to Argon2i-B.
3. The lower bound from Theorem 5 might be tight. Alwen and Blocki [AB16] gave an attack $\mathcal{A}$ such that $\Pi_{cc}^{\parallel}(\mathcal{A}) = O\left(n^{1.75}\delta \log n\right)$ for a (n, δ, t)-random DAG. In the following section we reduce the gap of $\tilde{O}\left(n^{1/12}\right)$ further by developing an improved recursive version of the attack of Alwen and Blocki [AB16]. In particular, we show that for any $\epsilon > 0$ we have $\Pi_{cc}^{\parallel}(\mathcal{A}) = o\left(n^{1+\sqrt{1/2}+\epsilon}\right) = o\left(n^{1.708}\right)$. Our modified attack also improves the upper bound for other iMHF candidates like Catena [FLW13].
4. Theorem 4 alone will not yield any meaningful lower bounds on the $\Pi_{cc}^{\parallel}$ of the Catena iMHFs [FLW13]. In particular, the results from Alwen and Blocki [AB16] imply that for any t-pass variant of Catena the corresponding DAG is (e, d)-reducible for $ed \geq nt$ (typically, $t = O(\mathsf{polylog}(n))$). However,

in the remainder of this section, we use an alternative techniques to prove that $\Pi^{\parallel}_{cc}(G) = \Omega\big(n^{1.5}\big)$ for the both Catena iMHFs and the Linear and DB iMHFs of [BCGS16].

5.2 Lowerbounding Dispersed Graphs

In this section we define dispersed graphs and prove a lowerbound on their CC. Next we show that several of the iMHF constructions from the literature are based on such graphs. Thus we obtain proofs of security for each of these constructions (albeit for limited levels of security). In the subsequent section we give an upperbound on the CC of these constructions showing that the lowerbounds in this section are relatively tight.

Generic Dispersed Graphs. Intuitively a (g, k)-dispersed DAG is a DAG ending with a path ϕ of length k which has widely dispersed dependencies. The following definitions make this concept precise.

Definition 5 (Dependencies). *Let $G = (V, E)$ be a DAG and $L \subseteq V$. We say that L has a* (z, g)-dependency *if there exist node disjoint paths $p_1, \ldots, p_z$ each ending in L and with length (at least) g.*

We are interested in graphs with long paths with many sets of such dependencies.

Definition 6 (Dispersed Graph). *Let $g \leq k$ be positive integers. A DAG G is called* (g, k)-dispersed *if there exists a topological ordering of its nodes such that the following holds. Let $[k]$ denote the final k nodes in the ordering of G and let $L_j = [jg, (j+1)g - 1]$ be the j^{th} subinterval. Then $\forall j \in [\lfloor k/g \rfloor]$ the interval L_j has a (g, g)-dependency.*

More generally, let $\epsilon \in (0, 1]$. If each interval L_j only has an $(\epsilon g, g)$-dependency then G is called (ϵ, g, k)-dispersed.

We show that many graphs in the literature consist of a *stack* of dispersed graphs. Our lowerbound on the CC of a dispersed graph grows in the height of this stack. The next definition precisely captures such stacks.

Definition 7 (Stacked Dispersed Graphs). *A DAG $G = (V, E)$ is called* $(\lambda, \epsilon, g, k)$-dispersed *if there exist $\lambda \in \mathbb{N}^+$ disjoint subsets of nodes $\{L_i \subseteq V\}$, each of size k with following two properties.*

1. *For each L_i there is a path running through all nodes of L_i.*
2. *Fix any topological ordering of G. For each $i \in [\lambda]$ let G_i be the sub-graph of G containing all nodes of G up to the last node of L_i. Then G_i is an (ϵ, g, k)-dispersed graph.*

We denote the set of $(\lambda, \epsilon, g, k)$-dispersed graphs by $\mathbb{D}^{\lambda,k}_{\epsilon,g}$.

We are now ready to state and prove the lowerbound on the CC of stacks of dispersed graphs.

Theorem 6.

$$G \in \mathbb{D}^{\lambda,k}_{\epsilon,g} \quad \Rightarrow \quad \Pi^{\parallel}_{cc}(G) \geq \epsilon\lambda g\left(\frac{k}{2} - g\right).$$

Intuitively we sum the CC of pebbling the last k nodes L_i of each sub-graph G_i. For this we consider any adjacent intervals A of $2g$ nodes in L_i. Let p be a path in the $(\epsilon g, g)$-dependency of the second half of A. Either at least one pebble is always kept on p while pebbling the first half of A (which takes time at least g since a path runs through L_i) or p must be fully pebbled in order to finish pebbling interval A (which also takes time at least g). Either way pebbling A requires an additional CC of g per path in the $(\epsilon g, g)$-dependency of the second half of A. Since there are $k/2g$ such interval pairs each with ϵg incoming paths in their dependencies we get a total cost for that layer of $kg\epsilon/2$. So the cost for all layer of G is at least $\lambda kg\epsilon/2$. The details (for the more general case when g doesn't divide n) can be found in Appendix B.

The Graphs of iMHFs. We apply Theorem 6 to some important iMHFs from the literature. For this we first describe the particular DAGs (or at least their salient properties) underlying the iMHF candidates for which we prove lower-bounds in this section. Then we state a theorem summarizing our lowerbounds for these graphs. Finally we prove the theorem via a sequence of lemma; one per iMHF being considered.

Catena Dragonfly. We begin with the Catena Dragonfly graph. We briefly recall the properties of the DFG^n_λ construction, relevant to our proof, summarized in the following lemma which follows easily from the definition of DFG^n_λ in [FLW13, Def. 8 & 9].

For this we describe the "bit-reversal" function (from which the underlying bit-reversal graph derives its name). Let $k \in \mathbb{N}^+$ such that $c = \log_2 k$ is an integer. On input $x \in [k]$ the *bit-reversal function* $\mathbf{br}(\cdot) : [k] \to [k]$ returns $y + 1$ such that the binary representation of $x - 1$ using c bits is the reverse of the binary representation of y using c bits.

Lemma 3 (Catena Dragonfly). *Let $\lambda, n \in \mathbb{N}^+$ be such that $k = n/(\lambda+1)$ is a power of 2. Let $G = \mathsf{DFG}^n_\lambda$ be the Catena Bit Reversal graph. Then the following holds:*

1. *G has n nodes.*
2. *Number them in topological order with the set $[n]$ and $\forall i \in [0, \lambda]$ let node set $L_i = [1 + ik, (i + 1)k]$. A path runs through all nodes in each set L_i.*
3. *Node $ki + x \in L_i$ has an incoming edge from $k(i - 1) + \mathbf{br}(x) \in L_{i-1}$.*

Catena Butterfly. Next describe the graph underlying the Catena Butterfly graph. We summarize its key properties relevant to our proof in the following lemma (which follows immediately by inspection of the Catena Butterfly definition [FLW13, Def. 10 & 11]).

Lemma 4 (Catena Butterfly Graph). *Let $\lambda, n \in \mathbb{N}^+$ such that $n = \bar{n}(\lambda(2c-1)+1)$ where $\bar{n} = 2^c$ for some $c \in \mathbb{N}^+$. Then the* Catena Butterfly Graph BFG^n_λ *consists of a stack of λ sub-graphs such that the following holds.*

1. *The graph BFG^n_λ has n nodes in total.*
2. *The graph BFG^n_λ is built as a stack of λ sub-graphs $\{G_i\}_{i\in[\lambda]}$ each of which is a superconcentrator[9]. In the unique topological ordering of BFG^n_λ denote the first and final $\bar{n}$ nodes of each G_i as $L_{i,0}$ and $L_{i,1}$ respectively. Then there is a path running through all nodes in each $L_{i,1}$.*
3. *Moreover, for any $i \in [\lambda]$ and subsets $S \subset L_{i,0}$ and $T \subset L_{i,1}$ with $|S| = |T| = h \leq \bar{n}$ there exist h node disjoint paths $p_1, \ldots, p_h$ of length $2c$ from S to T.*

Balloon Hashing Linear. Finally we describe the graph underlying both the Linear and DB construction [BCGS16]. The graph $G = \mathsf{Lin}^\sigma_\tau$ is a pseudo-randomly constructed τ-layered graph with $\mathsf{indeg}(G) = 21$. It is defined as follows:

- $G = (V, E)$ has $n = \sigma\tau$ nodes $V = [n]$, and G contains a path $1, 2, \ldots, n$ running through V.
- For $i \in [0, \tau - 1]$ let $L_i = [i\sigma + 1, (i+1)\sigma]$ denote the i'th layer. For each node $x \in L_i$, with $i > 0$, we select 20 nodes $y_1, \ldots, y_{20} \in L_{i-1}$ (uniformly at random) and add the directed edges $(y_1, x), \ldots, (y_{20}, x)$ to E.

5.3 The Lowerbounds.

Now that we have our lowerbound for stacks of dispersed graphs it remains to analyse for which parameters each of the above three graphs can be viewed as being dispersed graphs. The results of this analysis are summarized in the theorem bellow.

Theorem 7. *[iMHF Constructions Based on Dispersed Graphs]*

- *If $\lambda, n \in \mathbb{N}^+$ such that $n = \bar{n}(\lambda(2c-1)+1)$ where $\bar{n} = 2^c$ for some $c \in \mathbb{N}^+$ then it holds that*

$$\mathsf{BFG}^n_\lambda \in \mathbb{G}_{n,3} \qquad \mathsf{BFG}^n_\lambda \in \mathbb{D}^{\lambda,\bar{n}}_{1,\lceil\sqrt{\bar{n}}\rceil} \qquad \Pi^{\parallel}_{cc}(\mathsf{BFG}^n_\lambda) = \Omega\left(\frac{n^{1.5}}{c\sqrt{c\lambda}}\right).$$

- *If $\lambda, n \in \mathbb{N}^+$ such that $k = n/(\lambda+1)$ is a power of 2 then it holds that*

$$\mathsf{DFG}^n_\lambda \in \mathbb{G}_{n,2} \qquad \mathsf{DFG}^n_\lambda \in \mathbb{D}^{\lambda,k}_{1,\lceil\sqrt{k}\rceil} \qquad \Pi^{\parallel}_{cc}(\mathsf{DFG}^n_\lambda) = \Omega\left(\frac{n^{1.5}}{\sqrt{\lambda}}\right).$$

- *If $\sigma, \tau \in \mathbb{N}^+$ such that $n = \sigma * \tau$ then with high probability it holds that*

$$\mathsf{Lin}^\sigma_\tau \in \mathbb{G}_{n,21} \qquad \mathsf{Lin}^\sigma_\tau \in \mathbb{D}^{\tau-1,\sigma}_{0.25,\sqrt{\sigma}/2} \qquad \Pi^{\parallel}_{cc}(\mathsf{Lin}^\sigma_\tau) = \Omega\left(\frac{n^{1.5}}{\sqrt{\tau}}\right).$$

[9] A superconcentrator is a DAG with m inputs and outputs such that any subset of $s \in [m]$ inputs and outputs are connected by s node disjoint paths.

The theorem is proven in the following three lemma bellow (one lemma per graph). We begin with the graph for Catena Dragonfly.

Lemma 5. *It holds that* $\mathsf{DFG}^n_\lambda \in \mathbb{D}^{\lambda,k}_{1,\sqrt{k}}$ *where* $k = \frac{n}{(\lambda+1)}$ *and* $\Pi^{\parallel}_{cc}(\mathsf{DFG}^n_\lambda) = \Omega\left(\frac{n^{1.5}}{\sqrt{\lambda}}\right)$.

Proof of Lemma 5. Let $G = \mathsf{DFG}^n_\lambda$ and set $k = n/(\lambda+1)$, $c = \log_2 k$ and $g = \sqrt{k}$. By construction c is an integer. For simplicity assume c is even and so $g \in \mathbb{N}^+$.[10] Number the nodes of G according to (the unique) topological order with the set $[0, n-1]$. It suffices to show that for all $i \in [\lambda]$ the sub-graph G_i consisting of nodes $[(i+1)k - 1]$ is (g, k)-dispersed (with probability $\epsilon = 1$). If this holds then Theorem 6 immediately implies that $\Pi^{\parallel}_{cc}(\mathsf{DFG}^n_\lambda) = \Omega\left(\frac{n^{1.5}}{\sqrt{\lambda}}\right)$.

Recall that G consists of layerls L_i of length k. For each $j \in [k/2g]$ let $L_{i,j}$ be the j^{th} interval of $2g$ nodes of L_i. Let $R_{i,j}$ be the second half of $L_{i,j}$. We will show that there are g node-disjoint paths each terminating in $R_{i,j}$ whose remaining nodes are all in layer L_{i-1}. Let node set $S_x = [s + y - (g-2), s + y]$ where $y = \mathbf{br}(x)$ and $s = (i-1)k$. The next three properties follow immediately from Lemma 3 and they imply the lemma.

– $\forall x \in R$ it holds that $S_x \subset L_{i-1}$.
– $\forall x \in R$ there is a path of length g going through the nodes of S_x and ending in x.
– $\forall$ distinct $x, x' \in R$ sets S_x and $S_{x'}$ are disjoint. $\square$

Next we turn to the Catena Dragonfly graph.

Lemma 6. *Let* $\lambda, n \in \mathbb{N}^+$ *such that* $n = \bar{n}(\lambda(2c-1)+1)$ *with* $\bar{n} = 2^c$ *for some* $c \in \mathbb{N}^+$. *It holds that* $\mathsf{BFG}^n_\lambda \in \mathbb{D}^{\lambda,\bar{n}}_{1,g}$ *for* $g = \lceil\sqrt{\bar{n}}\rceil$ *and* $\Pi^{\parallel}_{cc}(\mathsf{BFG}^n_\lambda) = O\left(\frac{n^{1.5}}{c\sqrt{c\lambda}}\right)$.

Proof of Lemma 6. Let $G = \mathsf{BFG}^n_\lambda$ and let $G_1, G_2, \ldots, G_\lambda$ be the sub-graphs of G described in Lemma 4. We will show that each G_i is $(g, \bar{n})$-dispersed for $g = \lfloor\sqrt{\bar{n}}\rfloor$. Fix arbitrary $i \in [\lambda]$ and L_1 be the last $\bar{n}$ nodes in the (the unique) topological ordering of G_i. We identify the nodes in L_1 with the set $\{1\} \times [\bar{n}]$ such that the second component follows their topological ordering. Let $\bar{g} = \lfloor\bar{n}/g\rfloor$ and for each $j \in [\bar{g}]$ let $L_{1,j} = \{\langle 1, jg + x\rangle : x \in [0, g-1]\}$. We will show that $L_{1,j}$ has a (g, g)-dependency.

Let L_0 be the first $\bar{n}$ nodes of G_i which we identify with the set $\{0\} \times [\bar{n}]$ (again with the second component respecting their topological ordering). Notice that for $n > 1$ and $g = \lfloor\sqrt{\bar{n}}\rfloor$ it holds that $g(g - 2c + 1) \le n$. Thus the set $S = \{\langle 0, i(g - 2c + 1)\rangle : i \in [g]\}$ is fully contained in L_0. Property (3) of Lemma 4 implies there exist g node disjoint paths from S to $L_{1,j}$ of length $2c$. In particular $L_{1,j}$ has a $(g, 2c)$-dependency.

We extend this to a (g, g)-dependency. Let path p, beginning at node $\langle 0, v\rangle \in S$, be a path in the $(g, 2c)$-dependency of $L_{1,j}$. Prepend to p the path traversing

$$(\langle 0, v - (g - 2c - 1)\rangle, \langle 0, v - (g - 2c - 2)\rangle, \ldots, \langle 0, v\rangle)$$

[10] The odd case is identical but with messy but inconsequential rounding terms.

to obtain a new path p^+ of length g. As this is a subinterval of L_0 property (2) of Lemma 4 implies this prefix path always exists. Moreover since any paths $p \neq q$ in a $(g, 2c)$-dependency of $L_{1,i}$ are node disjoint they must, in particular, also begin at distinct nodes $\langle 0, v_p\rangle \neq \langle 0, v_q\rangle$ in S. But by construction of S any such pair of nodes is separated by $g - 2c$ nodes. In particular paths p^+ and q^+ are also node disjoint and so by extending all paths in a $(2c, g)$-dependency we obtain a (g, g)-dependency for $L_{1,i}$. This concludes the first part of the lemma.

It remains to lowerbound $\Pi^{\parallel}_{cc}(\mathsf{BFG}^n_\lambda)$ using Theorem 6.

$$\begin{aligned}\Pi^{\parallel}_{cc}(\mathsf{BFG}^n_\lambda) &\geq \lambda g\left(\frac{k}{2} - g\right) \geq \lambda\left\lfloor\sqrt{\bar{n}}\right\rfloor\left(\frac{\bar{n}}{2} - \left\lfloor\sqrt{\bar{n}}\right\rfloor\right) \\ &= \lambda\sqrt{\bar{n}}\left(\frac{\bar{n}}{2} - \sqrt{\bar{n}}\right) - O(\bar{n}) = \Omega\left(\lambda\bar{n}^{1.5}\right) \\ &= \Omega\left(\frac{n^{1.5}}{c\sqrt{c\lambda}}\right).\end{aligned}$$

□

Finally we prove a lowerbound for the Linear and DB variants of Balloon Hashing.

Lemma 7. *If $\sigma, \tau \in \mathbb{N}^+$ such that $n = \sigma\tau$ then with high probability it holds that*

$$\mathsf{Lin}^\sigma_\tau \in \mathbb{G}_{n,21} \qquad \mathsf{Lin}^\sigma_\tau \in \mathbb{D}^{\tau-1,\sigma}_{0.25,\sqrt{\sigma}/2} \qquad \Pi^{\parallel}_{cc}(\mathsf{Lin}^\sigma_\tau) = \Omega\left(\frac{n^{1.5}}{\sqrt{\tau}}\right).$$

Proof. (sketch) It suffices to show that $\mathsf{Lin}^\sigma_\tau \in \mathbb{D}^{\tau-1,\sigma}_{0.25,\sqrt{\sigma}/2}$. By Theorem 6 it immediately follows that

$$\Pi^{\parallel}_{cc}(\mathsf{Lin}^\sigma_\tau) \geq \frac{(\tau-1)\sqrt{\sigma}/2}{4}\left(\sigma/2 - \sqrt{\sigma}/2\right) = \Omega\left(\frac{n^{1.5}}{\sqrt{\tau}}\right) .$$

Fix any $i \in [0, \tau-1]$. Consider layer L_i and given set $S_x = [x, x+\sqrt{\sigma}/2-1] \subset L_i$ denoting an interval of $\sqrt{\sigma}/2$ nodes in L_i begining at node x. Without loss of generality we suppose that each node in S_x only has one randomly chosen parent in L_{i-1} — adding additional edges can only improve dispersity. We partition L_{i-1} into $\sqrt{\sigma}$ intervals of length $\sqrt{\sigma}$. We say that an interval $[u, u + \sqrt{\sigma} - 1] \subset L_{i-1}$ is *covered by* S_x if there is exists edge (y, v) from the second half of the interval to a node in S_x; that is if $y \in [u + \sqrt{\sigma}/2, u + \sqrt{\sigma} - 1]$ and $v \in S_x$. In this case the path $(u, u + 1, \ldots, y, v)$ has length $\geq \sqrt{\sigma}/2$ and this path will not intersect the corresponding paths from any of the other (disjoint) intervals in L_{i-1} (recall that we are assuming that $v \in S_x$ only has one parent in L_{i-1}). The probability that an interval $[u, u + \sqrt{\sigma} - 1] \subset L_{i-1}$ is covered by S_x is at least

$$1 - \left(1 - \frac{\sqrt{\sigma}/2}{\sigma}\right)^{\sqrt{\sigma}} \approx 1 - \sqrt{1/e} .$$

Thus, in expectation we will have at least $\mu = \sqrt{\sigma}\left(1 - \sqrt{1/e}\right) \geq 0.39 \times \sqrt{\sigma}$ node disjoint paths of length $\sqrt{\sigma}/2$ ending in S_x. Standard concentration bounds imply that we will have at least $\sqrt{\sigma}/4$ such paths with high probability. $\square$

6 New Memory-Efficient Evaluation Algorithm and Applications

In this section we introduce a new generic parametrized pebbling algorithm for DAGs (i.e. an evaluation algorithm for an arbitrary iMHF). We upperbound the pebbling strategy's cumulative pebbling complexity in terms of its parameters. In particular we see that for graphs which are not depth-robust there exist parameter settings for which the algorithm results in low CC pebbling strategies. Next we instantiate the parameters to obtain attacks on the random graphs defined in the previous section. By "attack" we mean that, for Argon2i-A and SB, the algorithm has significantly less asymptotic memory-hardness in the PROM than both that of their naïve algorithms, and even that of the attack in [AB16].

Review of [AB16]. In order to describe the results in this section we first review the generic pebbling algorithm $\mathsf{PGenPeb}$ of [AB16] which produces a pebbling $P_1, P_2, \ldots, P_n$ of G as follows. $\mathsf{PGenPeb}$ takes as input a node set $S \subset V$ of size $|S| = e$ such that removing S reduces the depth of the DAG $\mathsf{depth}(G - S) \leq d$. Intuitively, keeping pebbles on S compresses G in the sense that G can now quickly be entirely (re)pebbled within d (parallel) steps. This is because when S is already pebbled then no remaining unpebbled path has length greater than d. Algorithm $\mathsf{PGenPeb}$ never removes pebbles from nodes in S and its goal is to always pebble node i at time i so as to finish in $n = \mathsf{size}(G)$ steps.[11] To ensure that parents of node i are all pebbled at time i algorithm $\mathsf{PGenPeb}$ sorts nodes in topological order and partitions them into consecutive intervals of g nodes (where $g \in [d, n]$ is another input parameter). Nodes in interval $I_c = [(c-1)g + 1, cg] \cap [n] \subset V$ are pebbled during "light phase" Λ_c which runs for g time steps. To ensure that the result is a legal pebbling, $\mathsf{PGenPeb}$ guarantees the following invariant $\mathcal{I}$: just before light phase Λ_c begins (i.e. at time $(c-1)g$) we have $X_{cg} = \mathsf{parents}(I_c) \cap [(c-1)g] \subset P_{(c-1)g}$ so that we begin Λ_c with all of the necessary pebbles. Now, in phase Λ_c algorithm $\mathsf{PGenPeb}$ simply places a pebble on node $i \in I_c$ at time i.

Notice that for $c = 1$, $X_1 = \emptyset$ and so $\mathcal{I}$ is trivially satisfied. Let $c > 1$. Partition X_{cg} into $X_{cg}^- = X_{cg} \cap [(c-1)g - d + 1]$ and $X_{cg}^+ = X_{cg} \setminus X_{cg}^-$. Since X_{cg}^+ is pebbled in the final d steps of light phase Λ_{c-1}, $\mathsf{PGenPeb}$ can simply not remove those until time step $(c-1)g$. In order to ensure that X_{cg}^- is pebbled at that time $\mathsf{PGenPeb}$ also runs a "balloon phase" B_{c-1} in parallel with the final d steps of Λ_{c-1}.[12] Intuitively in phase B_{c-1} all nodes in $[(c-1)g] \subseteq V$ are quickly "decompressed" by greedily re-pebbling everything possible in parallel. Recall that pebbles are never removed from nodes in S. So at time j all of $S \cap [j]$ is

[11] Formally, $i \in P_i$ and $S \cap [i] \subseteq P_i$ for each $i \leq n$.

[12] Recall that $g \geq d$ so Λ_{c-1} lasts long enough to accommodate B_{c-1}.

already pebbled. Therefore, at time $(c-1)g$, there is no unpebbled path longer than d nodes within the first $[(c-1)g]$ nodes and so B_{c-1} can indeed entirely (and legally) repebble those nodes (and so in particular X_{cg}^{-}). Thus, together with the nodes in X_{cg}^{+} pebbled in the final d steps of Λ_{c-1} it follows that $\mathcal{I}$ also holds for Λ_c.

The runtime of $\mathsf{PGenPeb}$ is n. Thus the cost is at most $\Pi_{cc}^{\parallel}(\mathsf{PGenPeb}) \leq en + \delta gn + \lceil n/g \rceil (dn)$ where $\delta = \mathsf{indeg}(G)$. The en term upper bounds the cost of always keeping pebbles on S, δgn bounds the cost of all light phases, and the third term upper bounds the cost of all balloon phases — each balloon phase costs at most dn and at most $\lceil n/g \rceil$ balloon phases are run.

Notice that (for constant δ) we would like to set $g \leq e$ so that the second term doesn't dominate the first. Conversely, to keep the number of (expensive) balloon phases at a minimum we also want g to be large. Therefore, as long as $e \geq d$, the asymptotically minimal complexity is obtained when $g = e$.

Recursive Attack: Intuition. Our new algorithm relies on the following key insight. Algorithm $\mathsf{PGenPeb}$ can actually pebble, not just the sink with the above complexity, but instead any target set $T \subseteq V$ simultaneously.[13] This more general view allows us to recast the task of the balloon phase as such a pebbling problem. The graph being pebbled is $G' = G - (S \cup [(c-1)g - d + 1])$ and the target set is X_c^{-}. So instead of implementing balloon phases with an expensive greedy pebbling strategy as in $\mathsf{PGenPeb}$ we can apply the same strategy as (the generalized version of) $\mathsf{PGenPeb}$ recursively. This is the approach of the new algorithm $\mathsf{RGenPeb}$ (c.f. Algorithm 1 in Appendix B). For this approach to work we need that not only is G (e,d)-reducible via some set S but that there is also a set S' of size $e' > e$ such that $\mathsf{depth}(G - S') = d' < d$. Only when these conditions can no longer be met do we have to resort to greedy pebbling for the balloon phases. As we show below, it turns out that $\mathsf{RGenPeb}$ leads to improved attacks compared to $\mathsf{PGenPeb}$ for the DAGs underlying key iMHFs like Argon2i, Catena and Balloon Hashing.

Outline. The remainder of this section has the following structure. First, in Lemma 8 we generalize the results of [AB16] to upperbound the CC of a graph by the cost of pebbling all light phases plus the CC of the pebblings problems solved by balloon phases. Next we define a generalization of (e,d)-reducible graphs called f-reducible graphs; namely graphs which are $(f(d),d)$-reducible for all $d \in [n]$. This allows us to state the main theorem of this section. It considers a certain class of functions f and upper bounds the complexity of $\mathsf{RGenPeb}$ on such f-reducible graphs using any number k levels of recursion. To apply the theorem to the iMHFs from the literature we prove Lemma 9 which describes the f-reducibility of their underlying DAGs. Thus we obtain the final corollary of the section describing new upperbounds on those iMHFs. At the end of this section we give a more detailed description of $\mathsf{RGenPeb}$ and the proof of Lemma 8.

[13] For example, in the final d steps of the execution one last balloon phase can be run to (re)pebble all of G including T at no added cost to the asymptotic complexity.

Generalizing [AB16]. In order to derive the new pebbling strategy we first generalize the results of [AB16]. Given a DAG $G = (V, E)$, node set $T \subseteq V$ and integer t we define $\mathcal{P}^{\parallel}_{G,T,t} \subseteq \mathcal{P}^{\parallel}_{G,T}$ to be the set of all parallel pebblings $(P_1, \ldots, P_z)$ of G such that $z \leq t$. Analogously we let $\Pi^{\parallel}_{cc}(G, T, t) = \min_{P \in \mathcal{P}^{\parallel}_{G,T,t}} \Pi^{\parallel}_{cc}(P)$.

We remark that if $\mathsf{depth}(G) = d$ then $\Pi^{\parallel}_{cc}(G, T, d) \leq dn$ since we can greedily pebble G in topological order in time $\mathsf{depth}(G)$. Lemma 8 provides an alternative upper bound on $\Pi^{\parallel}_{cc}(G, T, 2d)$.

Lemma 8. *Let $G = (V, E)$ be a DAG of size n, indegree δ and $\mathsf{depth}(G) \leq d_0$. If G is (e_1, d_1)-reducible with parameters e_1, d_1 such that $2d_1 n \leq e_1 d_0$ and $d_1 \leq d_0$ then for any target set $T \subseteq V$ we have*

$$\Pi^{\parallel}_{cc}(G, T, 2d_0) \leq (4\delta + 4)\, e_1 d_0 + \frac{n}{e_1} \left(\max_{\substack{T' \subseteq V - S_1 \\ |T'| \leq \delta \cdot e_1}} \Pi^{\parallel}_{cc}\left(G - S_1, T', 2d_1\right) \right) ,$$

where $S_1 \subseteq V$ has size $|S_1| \leq e_1$ such that $\mathsf{depth}\left(G - S_1\right) \leq d_1$.

To prove the lemma we first define the RGenPeb algorithm and argue the legality of the pebbling it produces at the end of this section. Armed with this, it remains only to upperbound the complexity of a call to RGenPeb in terms of the complexity of the recursive call it makes. This involves a relatively straightforward (but somewhat tedious) counting of the pebbles placed by RGenPeb, the details of which can be found in Appendix B.

We observe that Lemma 8 generalizes the main result of [AB16] as that work only considered the special case where balloon phases are implemented with a greedy pebbling strategy. The advantage of the above formulation (and the more general RGenPeb) is that now we can be apply the lemma (and algorithm) recursively.

In order to apply this lemma repeatedly we will need graphs which are reducible for a sequence of points parameters (e, d) satisfying the conditions laid out in Lemma 8 relating consecutive parameters. To help characterize such graphs we generalize the notion of reducibility as follows.

Definition 8. *Let $G = (V, E)$ be a DAG with n nodes and let $f : \mathbb{N} \to \mathbb{N}$ be a function. We say that G is f-reducible if for every positive integer $n \geq d > 0$ there exists a set $S \subseteq V$ of $|S| = f(d)$ nodes such that $\mathsf{depth}(G - S) \leq d$.*

Next we state the main theorem of this section which, for a certain class of natural functions f, upperbounds the CC of any f-reducible graph.

Theorem 8. *Let G be a f-reducible DAG on n nodes then if $f(d) = \tilde{O}\left(\frac{n}{d^b}\right)$ for some constant $0 < b \leq 2/3$ and let $a = \frac{1-2b+\sqrt{1+4b^2}}{2}$. Then for any constant $\epsilon > 0$*

$$\Pi^{\parallel}_{cc}(G) \leq O\left(n^{1+a+\epsilon}\right) .$$

The proof of Theorem 8 is in the appendix. We briefly sketch the intuition here. We define a sequence $e_1, e_2, \ldots$ and $d_1, d_2, \ldots$ such that G is (e_i, d_i)-reducible for each i, $e_i = n^{a_i+\epsilon/3}$ and $d_i = n^{\frac{1-a_i}{b}}$ with

$$a_{i+1} = 1 + \frac{(a-1)(1-a_i)}{b}, \text{ where} \qquad a_1 = a = \frac{1-2b+\sqrt{1+4b^2}}{2} .$$

If $b \le a$ we have $e_{i+1}d_i \ge nd_{i+1}$ for every i so we can repeatedly invoke Lemma 9 as many times as we desire. By exploiting several key properties of the sequence $\{a_i\}_{i=1}^{\infty}$ we can show that unrolling the recurrence k times yields a pebbling with cost at most $k\,(4\delta+2)\,n^{1+a+\epsilon/3} + n^{1+a+\epsilon/3}d_k$. For any $\epsilon > 0$ we can select the constant k sufficiently large that $d_k \le n^{\epsilon/3}$. Thus, the pebbling cost is $o\left(n^{1+a+\epsilon}\right)$.

Analysing Existing iMHFs. We can now turn to applying Theorem 8 to iMHFs from the literature. Lemma 9 below states that an (n, δ, n)-random DAGs and λ-layered DAGs are f-reducible. In particular these are the types of DAGs underlying all of the iMHFs considered in the previous section.

Lemma 9. *Let* $f_b(d) = \tilde{O}\left(\frac{n}{d^b}\right)$ *then*

1. *Let* $\delta = O(\mathsf{polylog}(n))$ *then a* (n, δ, n)*-random DAG is* $f_{0.5}$*-reducible with high probability.*
2. *The Catena DAGs* DFG_λ^n *and* BFG_λ^n *are both* f_1*-reducible for* $\lambda = O(\mathsf{polylog}(n))$.
3. *The Balloon Hashing Linear (and the DB) graph* Lin_τ^σ *is* f_1*-reducible for* $\tau = O(\mathsf{polylog}(n))$.

The proof generalizes the arguments used in [AB16] to first establish a particular pair (e, d) for which the graphs are reducible. It can be found in Appendix B.

Together with Theorem 8 and Lemma 9 we now obtain the main application of $\mathsf{RGenPeb}$ which is described in the following corollary upperbounding the memory-hardness of each of the considered iMHFs.

Corollary 4. *Let* $\epsilon > 0$ *be any constant*

1. *Let* $\delta = O(\mathsf{polylog}(n))$ *then an* (n, δ, n)*-random DAG* G *has* $\Pi_{cc}^{\parallel}(G) = O\left(n^{1+\sqrt{1/2}+\epsilon}\right) \approx O\left(n^{1.707+\epsilon}\right)$.
2. *Both* $\Pi_{cc}^{\parallel}\left(\mathsf{DFG}_\lambda^n\right)$ *and* $\Pi_{cc}^{\parallel}\left(\mathsf{BFG}_\lambda^n\right)$ *are in* $\tilde{O}\left(n^{\frac{13}{8}}\right) = \tilde{O}\left(n^{1.625}\right)$.
3. $\Pi_{cc}^{\parallel}\left(\mathsf{Lin}_\tau^\sigma\right) = \tilde{O}\left(n^{\frac{13}{8}}\right) = \tilde{O}\left(n^{1.625}\right)$*, where* Lin_τ^σ *has* $n = \tau\sigma$ *nodes.*

We remark that Theorem 8 does not yield tighter bounds for Catena iMHFs DFG_λ^n or BFG_λ^n or for Lin_τ^σ. Each DAG is indeed f_b reducible for any $b \le 2/3$ (even for $b \le 1$), but for $b \le 2/3$ it follows that $a = \frac{1-2b+\sqrt{1+4b^2}}{2} \ge 2/3$. Thus, Theorem 8 yields an attack with cost $O\left(n^{\frac{5}{3}+\epsilon}\right)$, which does not improve on the non-recursive $\mathsf{PGenPeb}$ attack in [AB16] as that has cost $O\left(n^{\frac{5}{3}}\right)$. However, we can set $e_1 = n^{5/8}, e_2 = n^{7/8}$ and exploit the fact that the DAGs are (e_i, d_i)-reducible with $d_i = \tilde{O}(n/e_i)$. Applying Lemma 8 twice we have $\Pi_{cc}^{\parallel}(G) =$

$O\left(e_1 n + \frac{n}{e_1}\left(e_2 d_1 + \frac{n}{e_2} n d_2\right)\right) = O\left(n^{13/8} + n^{3/8+7/8} d_1 + n^{3/8+1/8+1} d_2\right) = \tilde{O}\left(n^{\frac{13}{8}}\right)$. Note that $e_2 d_1 = \tilde{O}\big(n^{10/8}\big) > \tilde{O}\big(n^{9/8}\big) = n d_2$ so it is legal to invoke Lemma 8 for sufficiently large n.

The RGenPeb *Algorithm.* In the remainder of this section we sketch RGenPeb algorithm and justify that it produces a legal pebbling. The analysis of its complexity in terms of the complexity of its recursive call is contained in the proof of Lemma 8. The final complexity of an execution requires unravelling the recursive statement of Lemma 8 which is done in the proof of Theorem 8.

In the following we will ignore rounding errors here as they are inconsequential for the asymptotic behaviour while adding needless complexity to the exposition. For completeness we observe that if RGenPeb finishes a light phase and there is not enough steps left to complete a full light phase then it can simply runs the next light phase as far as it can (and completely omits any further balloon phases). This affects neither the legality of the resulting pebbling nor its asymptotic complexity.

Algorithm RGenPeb takes input a DAG $G = (V, E)$, sets $S_1 \subseteq S_2 \subseteq \ldots \subseteq S_k \subseteq V$, integers $d_1, d_2, \ldots, d_k$ and a target set T such that $\forall i \in [k] : e_i d_{i-1} \geq n d_i$ and $d_i \geq \mathsf{depth}(G - S_i)$ where $n = |V|$, $e_i = |S_i|$ and $d_0 = \mathsf{depth}(G)$. For this RGenPeb makes use of an arbitrary partition of the nodes of G into $2d_0$ into sets $D_1, D_2, \ldots, D_{2d_0}$ such that the following properties hold:[14]

Topologically Ordered: $\forall j \in [2d_0 - 1] \;\; \mathsf{parents}(D_{j+1}) \subseteq \bigcup_{y \in [j]} D_y$,
Maximum Size: $\forall j \leq 2d_0 \;\; |D_j| \leq \frac{n}{d_0}$.

Intuitively, the set D_j is the set of nodes that will be pebbled by a light phase in the j^{th} step. So for PGenPeb we would simply have $D_j = \{j\}$.

At the top level of the recursion RGenPeb looks relatively similar to PGenPeb. The goal is to pebble $G_0 = G$ with the target set $T_0 = \mathsf{sinks}(G_0)$ in at most $2d_0$ steps which is done by executing a sequence of light phases lasting $m = e_2 d_0/n$ steps and balloon phases lasting $2d_1$ steps. The requirement that $e_2 d_0 \geq 2 d_1 n$ ensures that $m \geq 2d_1$ so that we can complete each balloon phase in time for the upcoming light phase. For $t \in [2d_0]$ let $U_t = \bigcup_{j \in [t]} D_j$ be all nodes pebbled by light phases up to step t. Then for $c \in [2d_0/m]$ the light phase Λ_c runs during time interval $I_c = [(c-1)m + 1, cm]$ during which it will pebble nodes $U_{cm} \setminus U_{(c-1)m}$. It never removes pebbles from S_1 and, at each time step t it keeps pebbles on $\mathsf{parents}(U_{cm} \setminus U_t)$ as it will still need those to finish the light phase.

As for PGenPeb the light phase Λ_1 is trivially a legal pebbling. Let $X_{cm} = \mathsf{parents}(U_{cm+m} \setminus U_{cm+1}) \cap U_{cm}$ and $X^-_{cm} = (X_{cm} \cap U_{cm-2d_1}) \setminus S_1$ and $X^+_{cm} = X_{cm} \setminus X^-_{cm}$. To ensure that all pebbles placed by during light phase Λ_{c+1} are done so legally it suffices for RGenPeb to ensure that X_{cm} is fully pebbled at time cm. This is done by balloon phase running in parallel to the final $2d_1$ steps of Λ_c; that is during the interval $[cm - 2d_1, cm]$. The pebbling for the balloon phase may be

[14] For example we can sort the nodes in topological order and divided them up into the partition. Whenever a set is larger than n/d_0 we insert a new set into the partition with the overflow.

obtained by a recursive call to RGenPeb for the graph $G' = G - S - (V \setminus U_{cm-2d_1}))$ (G' is the DAG induced by nodes $U_{cm-2d_1} - S$) with target set X_{cm}^- as well as parameters $S_2 \subseteq S_3 \subseteq \ldots \subseteq S_k$ and $d_2, d_3, \ldots, d_k$ (both lists now have length $k-1$ and clearly still satisfy the conditions on parameters stated above). If RGenPeb is ever called with empty lists $\bar{S} = \emptyset$ and $\bar{d} = \emptyset$ (i.e., $k = 0$) then it simply greedy pebbles G. The result of the recursive call is added to the final $2d_1$ steps of light phase Λ_c. Finally the pebbling is modified to never remove pebbles from X_{cm} during the those final steps of Λ_{c-1}. Notice that each node in X_{cm}^+ is either in S or is pebbled at some point during the final $2d_1$ steps of Λ_{c+1}. Thus we are guaranteed that $X_{cm} \subseteq P_{(c-1)m}$ as desired.

To see why RGenPeb produces a legal pebbling it suffices to observe that pebbles placed during light phases always have their parents already pebbled. So if the recursive call returns a legal pebbling for the balloon phase then the final result is also legal. But at the deepest level of the recursion RGenPeb resorts to a greedy pebbling which is trivially legal. Thus, by induction, so is the pebbling at the highest level of the recursion.

7 Open Questions

We conclude with several open questions for future research.

- We showed that for some constant $c \geq 0$ we can find a DAG G on n nodes with $\Pi_{cc}^{\parallel}(G) \geq cn^2/\log(n)$ and $\mathsf{indeg}(G) = 2$. While this result is asymptotically optimal the constant terms are relevant for practical applications to iMHFs. How big can this constant c be? Can we find explicit constructions of constant-indegree, $(c_1 n/\log(n), c_2 n)$-depth robust DAGs that match these bounds?
- Provide tighter upper and lower bounds on $\Pi_{cc}^{\parallel}(G)$ for Argon2i-B [BDKJ16], the most recent version of Argon2i which was submitted to IRTF for standardization.
- Another interesting direction concerns understanding the cumulative pebbling complexity of generic graphs. Given a graph G is it computationally tractable to (approximately) compute $\Pi_{cc}^{\parallel}(G)$? An efficient approximation algorithm for $\Pi_{cc}^{\parallel}(G)$ would allow us to quickly analyze candidate iMHF constructions. Conversely, as many existing iMHF constructions are based on fixed random graphs, [BDK16,BCGS16] showing that approximating such a graphs complexity is hard would provide evidence that an adversary will likely not be able to leverage properties of the concrete instance to improve their evaluation strategy for the iMHF. Indeed, it may turn out that the most effective way to construct depth-robust graphs with good constants is via a randomized construction.

Acknowledgments

The authors would like to thank Pierrick Gaudry for his careful reading and many helpful suggestions. The first and third authors were supported by the ERC starting grant (259668-PSPC).

References

AB16. Joël Alwen and Jeremiah Blocki. Efficiently Computing Data-Independent Memory-Hard Functions. In *Advances in Cryptology CRYPTO'16*, pages 241–271. Springer, 2016.

AB17. Joël Alwen and Jeremiah Blocki. Towards Practical Attacks on Argon2i and Balloon Hashing. In *Proceedings of the 2nd IEEE European Symposium on Security and Privacy (EuroS&P 2017)*, page (to appear). IEEE, 2017. http://eprint.iacr.org/2016/759.

ABW03. Martín Abadi, Michael Burrows, and Ted Wobber. Moderately hard, memory-bound functions. In *Proceedings of the Network and Distributed System Security Symposium, NDSS 2003, San Diego, California, USA*, 2003.

ACK+16. Joël Alwen, Binyi Chen, Chethan Kamath, Vladimir Kolmogorov, Krzysztof Pietrzak, and Stefano Tessaro. On the complexity of `scrypt` and proofs of space in the parallel random oracle model. In *Advances in Cryptology - EUROCRYPT 2016 - Vienna, Austria, May 8-12, 2016, Proceedings, Part II*, pages 358–387, 2016. http://eprint.iacr.org/2016/100.

ACP+17. Joël Alwen, Binyi Chen, Krzysztof Pietrzak, Leonid Reyzin, and Stefano Tessaro. `scrypt` is Maximally Memory-Hard. In *Advances in Cryptology-EUROCRYPT 2017*, page (to appear). Springer, 2017. http://eprint.iacr.org/2016/989.

AGK+16. Jol Alwen, Peter Gai, Chethan Kamath, Karen Klein, Georg Osang, Krzysztof Pietrzak, Leonid Reyzin, Michal Rolnek, and Michal Rybr. On the memory-hardness of data-independent password-hashing functions. Cryptology ePrint Archive, Report 2016/783, 2016. http://eprint.iacr.org/2016/783.

AS15. Joël Alwen and Vladimir Serbinenko. High Parallel Complexity Graphs and Memory-Hard Functions. In *Proceedings of the Eleventh Annual ACM Symposium on Theory of Computing*, STOC '15, 2015. http://eprint.iacr.org/2014/238.

BCGS16. Dan Boneh, Henry Corrigan-Gibbs, and Stuart Schechter. Balloon hashing: Provably space-hard hash functions with data-independent access patterns. Cryptology ePrint Archive, Report 2016/027, Version: 20160601:225540, 2016. http://eprint.iacr.org/.

BDK15. Alex Biryukov, Daniel Dinu, and Dmitry Khovratovich. Fast and tradeoff-resilient memory-hard functions for cryptocurrencies and password hashing. Cryptology ePrint Archive, Report 2015/430, 2015. http://eprint.iacr.org/2015/430.

BDK16. Alex Biryukov, Daniel Dinu, and Dmitry Khovratovich. Argon2 password hash. Version 1.3, 2016. https://www.cryptolux.org/images/0/0d/Argon2.pdf.

BDKJ16. Alex Biryukov, Daniel Dinu, Dmitry Khovratovich, and Simon Josefsson. The memory-hard Argon2 password hash and proof-of-work function.

Internet-Draft draft-irtf-cfrg-argon2-00, Internet Engineering Task Force, March 2016.

BK15. Alex Biryukov and Dmitry Khovratovich. Tradeoff cryptanalysis of memory-hard functions. Cryptology ePrint Archive, Report 2015/227, 2015. `http://eprint.iacr.org/`.

Cha73. Ashok K. Chandra. Efficient compilation of linear recursive programs. In *SWAT (FOCS)*, pages 16–25. IEEE Computer Society, 1973.

Coo73. Stephen A. Cook. An observation on time-storage trade off. In *Proceedings of the Fifth Annual ACM Symposium on Theory of Computing*, STOC '73, pages 29–33, New York, NY, USA, 1973. ACM.

DFKP15. Stefan Dziembowski, Sebastian Faust, Vladimir Kolmogorov, and Krzysztof Pietrzak. Proofs of space. In Rosario Gennaro and Matthew J. B. Robshaw, editors, *CRYPTO 2015, Part II*, volume 9216 of *LNCS*, pages 585–605. Springer, Heidelberg, August 2015.

DGN03. Cynthia Dwork, Andrew Goldberg, and Moni Naor. On memory-bound functions for fighting spam. In *Advances in Cryptology - CRYPTO 2003*, volume 2729 of *Lecture Notes in Computer Science*, pages 426–444. Springer, 2003.

DKW11a. Stefan Dziembowski, Tomasz Kazana, and Daniel Wichs. Key-evolution schemes resilient to space-bounded leakage. In Phillip Rogaway, editor, *CRYPTO 2011*, volume 6841 of *LNCS*, pages 335–353. Springer, Heidelberg, August 2011.

DKW11b. Stefan Dziembowski, Tomasz Kazana, and Daniel Wichs. One-time computable self-erasing functions. In Yuval Ishai, editor, *TCC*, volume 6597 of *Lecture Notes in Computer Science*, pages 125–143. Springer, 2011.

DNW05. Cynthia Dwork, Moni Naor, and Hoeteck Wee. Pebbling and proofs of work. In Victor Shoup, editor, *CRYPTO 2005*, volume 3621 of *LNCS*, pages 37–54. Springer, Heidelberg, August 2005.

EGS75. Paul Erdoes, Ronald L. Graham, and Endre Szemeredi. On sparse graphs with dense long paths. Technical report, Stanford, CA, USA, 1975.

FLW13. Christian Forler, Stefan Lucks, and Jakob Wenzel. Catena: A memory-consuming password scrambler. *IACR Cryptology ePrint Archive*, 2013:525, 2013.

HJO$^+$16. Brett Hemenway, Zahra Jafargholi, Rafail Ostrovsky, Alessandra Scafuro, and Daniel Wichs. Adaptively secure garbled circuits from one-way functions. In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part III*, volume 9816 of *LNCS*, pages 149–178. Springer, Heidelberg, August 2016.

HP70. Carl E. Hewitt and Michael S. Paterson. Record of the Project MAC Conference on Concurrent Systems and Parallel Computation. chapter Comparative Schematology, pages 119–127. ACM, New York, NY, USA, 1970.

JW16. Zahra Jafargholi and Daniel Wichs. Adaptive Security of Yao's Garbled Circuits. Cryptology ePrint Archive, Report 2016/814, 2016. `http://eprint.iacr.org/2016/814`.

Kal00. Burt Kaliski. Pkcs# 5: Password-based cryptography specification version 2.0. 2000.

LT82. Thomas Lengauer and Robert E. Tarjan. Asymptotically tight bounds on time-space trade-offs in a pebble game. *J. ACM*, 29(4):1087–1130, October 1982.

MMV13. Mohammad Mahmoody, Tal Moran, and Salil P. Vadhan. Publicly verifiable proofs of sequential work. In Robert D. Kleinberg, editor, *ITCS 2013*, pages 373–388. ACM, January 2013.

NBF+15. Arvind Narayanan, Joseph Bonneau, Edward W Felten, Andrew Miller, and Steven Goldfeder. *Bitcoin and Cryptocurrency Technology (manuscript)*. 2015. Retrieved 8/6/2015.

Per09. C. Percival. Stronger key derivation via sequential memory-hard functions. In *BSDCan 2009*, 2009.

PHC. Password hashing competition. `https://password-hashing.net/`.

PR80. Wolfgang J. Paul and Rüdiger Reischuk. On alternation II. A graph theoretic approach to determinism versus nondeterminism. *Acta Inf.*, 14:391–403, 1980.

RD16. Ling Ren and Srinivas Devadas. Proof of space from stacked bipartite graphs. Cryptology ePrint Archive, Report 2016/333, 2016. `http://eprint.iacr.org/`.

Sch82. Georg Schnitger. A family of graphs with expensive depth reduction. *Theor. Comput. Sci.*, 18:89–93, 1982.

Sch83. Georg Schnitger. On depth-reduction and grates. In *24th Annual Symposium on Foundations of Computer Science, Tucson, Arizona, USA, 7-9 November 1983*, pages 323–328. IEEE Computer Society, 1983.

SS78. John E. Savage and Sowmitri Swamy. Space-time trade-offs on the fft algorithm. *IEEE Transactions on Information Theory*, 24(5):563–568, 1978.

SS79a. John E. Savage and Sowmitri Swamy. Space-time tradeoffs for oblivious interger multiplications. In Hermann A. Maurer, editor, *ICALP*, volume 71 of *Lecture Notes in Computer Science*, pages 498–504. Springer, 1979.

SS79b. Sowmitri Swamy and John E. Savage. Space-time tradeoffs for linear recursion. In Alfred V. Aho, Stephen N. Zilles, and Barry K. Rosen, editors, *POPL*, pages 135–142. ACM Press, 1979.

Tom78. Martin Tompa. Time-space tradeoffs for computing functions, using connectivity properties of their circuits. In *Proceedings of the Tenth Annual ACM Symposium on Theory of Computing*, STOC '78, pages 196–204, New York, NY, USA, 1978. ACM.

Val77. Leslie G. Valiant. Graph-theoretic arguments in low-level complexity. In Jozef Gruska, editor, *Mathematical Foundations of Computer Science 1977, 6th Symposium, Tatranska Lomnica, Czechoslovakia, September 5-9, 1977, Proceedings*, volume 53 of *Lecture Notes in Computer Science*, pages 162–176. Springer, 1977.

A Memory-Hard Functions

We define MHFs in the Parallel Random Oracle Model (pROM) of [AS15]. For this we first define the model and associated complexity notions and then fix the exact notion of MHF.

The Parallel Random Oracle Model. We consider an arbitrary repeatedly invoked algorithm $\mathcal{A}$ executing in the parallel random oracle model (pROM) [AS15] of computation where we make states between invocations explicit as follows. At invocation $i \in \{1, 2, \ldots\}$ algorithm $\mathcal{A}$ is given the state (bit-string) σ_{i-1} it produced at the end of the previous invocation. Next $\mathcal{A}$ can make a batch of calls $\mathbf{q}_i = (q_{1,i}, q_{2,i}, \ldots)$ to the *fixed input length* random oracle H (a.k.a. an ideal compression function). Then it receives the response from H and can perform arbitrary computation before finally outputting an updated state σ_i. The initial state σ_0 contains the input to the computation which terminates once a special final state is produced by $\mathcal{A}$. Apart from the explicit states σ the algorithm may keep no other state between invocations. For an input x and coins r we denote by $\mathcal{A}(x; r; H)$ the corresponding (deterministic) execution of $\mathcal{A}$. We say that $\mathcal{A}$ is *sequential* if in no execution does it ever make a batch of queries $\mathbf{q}$ with $|\mathbf{q}| > 1$.

The *cumulative memory complexity* (CMC) is defined to be

$$\mathsf{cmc}(\mathcal{A}) = \mathop{\mathbb{E}}_{H}\left[\max_{x,r} \sum_i |\sigma_i|\right]$$

where $|\sigma|$ is the bit-length of state σ, the expectation is taken over the choice of H and $\max_{x,r}$ denotes the maximum over all inputs and coins of $\mathcal{A}$.

We also need the following two worst-case complexity notions. The *time complexity* (TC) $\mathsf{time}(\mathcal{A})$ is the maximum running time of $\mathcal{A}$ in any execution (over all choices of x, r and H). Similarly, the *space complexity* (SC) is the largest state it ever outputs in any execution.

$$\mathsf{space}(\mathcal{A}) = \max_{x,r,H} \{\max_i |\sigma_i|\}.$$

We remark that SC and TC are somewhat stricter than usual since we maximise over all choices of H. However, this can only help us as these measures are used as worst case estimates of the complexity for the honest party and we ask that an MHF has reasonable upper-bounds on their values.

An oracle function f is a function over strings which depends on the choice of H. Let $\mathbb{A}_{f,m,q}$ be the set of pROM algorithms which compute f on $m \in \mathbb{N}^+$ arbitrary (distinct) inputs making at most q queries to H. Then the *amortized cumulative memory complexity* (aCMC) of f is defined to be

$$\mathsf{cmc}_{m,q}(f) = \min\left\{\frac{\mathsf{cmc}(\mathcal{A})}{n} \ : \ n \in [m], \mathcal{A} \in \mathbb{A}_{f,n,q}\right\}.$$

We comment on two differences between the above complexity notions and those in [AS15] (and why they do not prevent us from using the results in that work).

- In the definition of CMC above we maximize over all coins of $\mathcal{A}$ instead of including them in the expectation. This is with out loss of generality for the aCMC of a function since hardcoding the coins which minimize the aCMC of any $\mathcal{A}$ has at least as much CMC as the expected value for random coins.
- The aCMC of [AS15] is also parametrized by the minimum success probability ϵ of any algorithm computing f. Instead, for reasons of exposition, in this work we restrict ourselves to the special case when $\epsilon = 1$.

Memory-Hard Functions. As observed in [AS15] the aCMC of a function provides a good lower-bound on the amortized AT-complexity of that function. Thus the following definition captures the intuition of a memory-hard function in the pROM.

Definition 9 (Memory-Hard Function). *Let $\{f_{\sigma,\tau}\}_{\sigma,\tau\in\mathbb{N}^+}$ be a family of (oracle) functions and $\mathcal{N}$ be a sequential pROM algorithm which, on input (σ,τ,x), outputs $f_{\sigma,\tau}(x)$ in time at most $\tau\sigma$ using space at most σ. Then $F = (\{f_{\sigma,\tau}\},\mathcal{N})$ is an (h,g,t)*-memory-hard function (for up to m instances and q queries) *if it has* memory-hardness *at least h,* memory-gap *at most g and* throughput *at least t (all functions of σ and τ).*

$$\mathsf{cmc}_{m,q}(f_{\sigma,\tau}) \geq h(\gamma,\tau) \qquad \frac{\mathsf{space}(\mathcal{N}) * \mathsf{time}(\mathcal{N})}{\mathsf{cmc}_{m,q}(f_{\sigma,\tau})} \leq g(\sigma,\tau) \qquad \frac{\mathsf{space}(\mathcal{N})}{\mathsf{time}(\mathcal{N})} \geq t(\sigma,\tau).$$

In practice, we may content ourselves with families of infinite size but which do not contain a member for possible $k \in \mathbb{N}^+$ as long as the family is not too sparse (e.g. we have a function for all powers of 2).

Using the following theorem from [AS15] the results in this work imply MHFs with various desirable properties.

Theorem 9 ([AS15]). *Let $G \in \mathbb{G}_{n,\delta}$, P be a sequential pebbling of G with $(\Pi_t(P),\Pi_s(P)) = (z_n, s_n - 1)$ and let H be a random oracle with $w > 13$ bits of output. Fix any $m,q \in \mathbb{N}^+$ subject to the following (reasonable) pair of constraints.*

- *Not too many copies of f are computed: $m \leq 2^{w-2}/n$.*
- *Not too many oracle queries are made: $q \leq 2^{w/2}$.*

Then there exists an oracle function f and pROM evaluation algorithm $\mathcal{N}$ for f with

$$\mathsf{time}(\mathcal{N}) = z_n \qquad \mathsf{space}(\mathcal{N}) = w * s_n \qquad \mathsf{cmc}_{m,q}(f) \geq \frac{w * \Pi^{\parallel}_{cc}(G)}{4}.$$

In particular this theorem shows that in order to construct an MHF with both memory-cost σ and time-cost τ parameters it suffices to construct a family of DAGs for every size (with high CC) together with matching sequential pebblings. For simplicity we state the theorem for DAGs with a single source and sink but it can be easily extended to the more general case.[15]

Corollary 5 (High CC Graphs Imply MHFs). *Let $\{G_n \in G_{n,\delta_n}\}_{n=1}^{\infty}$ be a family of DAGs each with a single source and sink and let H be a random oracle with $w > 13$ bits of output. For each n let P_n be a sequential pebbling of G_n where $(\Pi_t(P_n),\Pi_s(P_n)) = (z_n, s_n - 1)$. Then there exists a (h,g,t)-MHF with*

$$h(\sigma,\tau) \geq \frac{w * \tau * \Pi^{\parallel}_{cc}(G_\sigma)}{4} \qquad g(\sigma,\tau) \leq \frac{4z_\sigma * s_\sigma}{\Pi^{\parallel}_{cc}(G_\sigma)} \qquad t(\sigma,\tau) \geq \frac{w * s_\sigma}{z_\sigma \tau}.$$

[15] Moreover any DAG can easily be extended to be of this form with no penalty to the CC as a function of its size.

Proof. The idea is simple. Fix any σ and τ. To obtain oracle function connect τ copies of the DAG G_σ in a chain and let $f_{\sigma,\tau}$ be the MHF given by Theorem 9. The corresponding sequential pebbling $P_{\sigma,\tau}$ is simply the pebbling P_σ repeated τ times for each copy of G_σ with the following caveat. Whenever a pebble is placed on the source node of any copy of G_σ it is only removed once no more children of that node will be pebbled. Thus $\mathsf{space}(P_{\sigma,\tau}) \leq \mathsf{space}(P_\sigma) + w = w * s_\sigma$ and $\mathsf{time}(P_{\sigma,\tau}) = \tau * \mathsf{time}(P_\sigma) = \tau z_\sigma$. Finally, it is easy to see that $\Pi^{\parallel}_{cc}(G_{\sigma,\tau}) = \tau * \Pi^{\parallel}_{cc}(G_\sigma)$ since any pebbling of $G_{\sigma,\tau}$ with CC less than $\tau * \Pi^{\parallel}_{cc}(G_\sigma)$ would have to pebble at least one copy of G_σ with less than $\Pi^{\parallel}_{cc}(G_\sigma)$ which is a contradiction.

Two remarks are in order.

- In practice one would probably want a stronger connection between copies of G_σ. For example one could connect the last σ nodes pebbled by P_σ in one copy of G_σ to the first σ nodes pebbled by P_σ in the next copy of G_σ. This would affect neither the time nor space complexity of the honest evaluation algorithm but would potentially increase the concrete (though not asymptotic) memory-hardness of $f_{\sigma,\tau}$ which would also result in a smaller memory-gap. For the purpose of this work though the simple construction in the proof suffices as it has the same asymptotic behaviour.
- Estimating the effect of requiring a pebble on the source of each internal copy of G_σ to potentially require the space complexity to grow by 1 is extremely pesimistic. To the best of our knowledge the $\mathcal{N}$ algorithm for all MHF constructions in the literature as well the sequential pebbling of all DAGs in this work already keep such a pebble there (rather than repebble the source repeatedly). Thus the space complexity of the sequential pebbling when composing those DAGs would not grow by 1. Never the less, rather than make the corollary seem less general then it is (by requiring such a property from P) we have opted for its current form. In particular the difference is both asymptotically, and practically speaking, imaterial.

B Missing Proofs

The proof of Corollary 1 uses the following result from [AB16] which shows that DAGs that are not depth-robust also have low $\Pi^{\parallel}_{cc}$.

Theorem 10 ([AB16, Thm. 2]). *Let $G_n \in \mathbb{G}_{n,\delta}$ such that G_n is not (e,d)-depth-robust. Then*

$$\Pi^{\parallel}_{cc}(G_n) = O\left(\min_{g\in[d,n]}\left\{n\left(\frac{dn}{g} + \delta g + e\right)\right\}\right)$$

setting $g = \sqrt{\frac{dn}{\delta}}$ this simplifies to $\Pi^{\parallel}_{cc}(G) = O\left(n(\sqrt{dn\delta} + e)\right)$.

Reminder of Corollary 1. *For some constants $c_1, c_2 > 0$ there exists an infinite family of DAGs $\{G_{n,\delta} \in \mathbb{G}_{n,\delta}\}_{n=1}^{\infty}$ with $\delta \leq c_1 \log(n)$ and $\Pi^{\parallel}_{cc}(G) \geq$*

c_2n^2. This is optimal in the sense that for any family $\{\delta_n \in [n]\}_{n=1}^{\infty}$ and $\{J_n \in \mathbb{G}_{n,\delta_n}\}_{n=1}^{\infty}$ it holds that $\Pi_{cc}^{\parallel}(J_n) \in O(n^2)$. Moreover if $\delta_n = o(\log(n)/\log\log(n))$ then $\Pi_{cc}^{\parallel}(J_n) = o(n^2) = o(\Pi_{cc}^{\parallel}(G_n))$.

Proof of Corollary 1. Take $\{G_n \in \mathbb{G}_{n,c_3\log(n)}\}_{n=1}^{\infty}$ to be the family of DAGs from Theorem 3. Now the first and second statements follow immediately from Theorem 4 and the observation that $\Pi_{cc}^{\parallel}(J_n) \leq n^2$ for any n node DAG J_n. The final statement is based on the simple observation that $\Pi_{cc}^{\parallel}(J_n) = o\big(n^2\big)$ whenever $\delta_n = o\big(\log(n)/\log\log(n)\big)$ because any such DAG is $(o(n), o(n/\delta_n))$-reducible. We can see this by applying Lemma 10, due to Valiant [Val77], $3\log(\delta_n)$ times to obtain a set S of at most

$$e = |S| \leq \frac{3\log(\delta_n)n\delta_n}{\log(n) - 2\log(\delta_n)} = o(n)$$

nodes such that $d = \mathsf{depth}(G) \leq 2^{-3\log(\delta_n)}n = o(n/\delta_n^2)$. Now by Theorem 10 we have $\Pi_{cc}^{\parallel}(G) = O(ne + n\sqrt{dn\delta_n}) = o(n^2)$. $\square$

Lemma 10 ([Val77] Extension). *Given a DAG G with m edges and depth $\mathsf{depth}(G) \leq d = 2^i$ there is a set of m/i edges s.t. by deleting them we obtain a graph of depth at most $d/2$.*

Reminder of Lemma 2. *For any $e \geq \sqrt{n}$ any any $\delta \geq 2$ a (n,δ,n)-random DAG will be $\left(e, \Omega\left(\frac{n^2}{e^2\log(n)}\right), \frac{n}{20e}\right)$-block depth robust except with negligible probability in n.*

Proof of Lemma 2. Let G be a random (n,δ,n)-random DAG $G = (V,E)$ with nodes $V = [n]$. Fix an arbitrary integer $m \in [n]$ and set $n' = \lfloor n/m \rfloor$. We will define a DAG G_m called the meta-graph of G. For this we use the following sets. For all $i \in [n']$ let $M_i = [(i-1)m+1, im] \subseteq V$. Moreover we denote the first and last thirds respectively of M_i with

$$M_i^F = [(i-1)m+1, (i-1)m + \lfloor m/3 \rfloor] \subseteq M_i\ ,$$

and

$$M_i^L = [(i-1)m + \lceil 2m/3 \rceil + 1, im] \subseteq M_i\ .$$

We define the meta-graph $G_m = (V_m, E_m)$ as follows:

Nodes: V_m contains one node v_i per set M_i. We call v_i the *simple node* and M_i its *meta-node*.

Edges: If the end of a meta-node M_i^L is connected to the beginning M_j^F of another meta-node we connect their simple nodes.

$$V_m = \{v_i : i \in [n']\} \qquad\qquad E_m = \{(v_i, v_j) : E \cap (M_i^L \times M_j^F) \neq \emptyset\}.$$

Lemma 2 now follows from the next two claims.

Claim 1 *If G_m is (e,d)-depth robust then G is $\big(e/2, dm/3, m\big)$-block depth robust.*

Proof. Fix any set $S \subseteq V$ of size $e/2$. We say that a node $v_i \in V_m$ in the meta-graph is unaffected by S if $M_i \cap N(S,m) = \emptyset$. That is $G - N(S,m)$ contains every node in the set M_i. Let $S_m \subseteq V_m$ denote the set of nodes affected by S. Formally, $S_m = \{v_i \in V_m \ : \ M_i \cap N(S,m) \neq \emptyset\}$.

We now claim that $|S_m| \leq e$. To see this we observe that the set

$$N(S,m) = \bigcup_{v \in S} \{v - m + 1, \dots, v\}$$

can intersect at most e meta-nodes because for each $v \in S$ the set $\{v - m + 1, \dots, v\}$ intersects at most two meta-nodes. Thus, S affects at most e nodes in G_m.

Since G_m is (e,d)-depth robust there remains a path ϕ' of length d in $G_m - S_m$. To complete the proof we observe that ϕ' corresponds to a path ϕ in $G - N(S,m)$ of length

$$\mathsf{length}(\phi) \geq \frac{\mathsf{length}(\phi')m}{3} \geq \frac{dm}{3}.$$

In particular, the path ϕ goes through the middle thirds of the d meta-nodes corresponding to ϕ'. Since each of corresponding meta-nodes in ϕ' is unaffected by S the path ϕ is still contained in $G - N(S,m)$. □

Claim 2 *Let $n^{0.01} < m < n^{0.5}$ then G_m is $\big(n/(10m), m/(200\log(n))\big)$-depth robust except with negligible probability in n.*

Proof. Divide G_m into $d = m/(100\log(n))$ layers $L_1, \dots, L_d$ each containing n'/d consecutive nodes in G_m. Given $i < j$ we say that layer L_i and L_j are connected by a γ-bipartite expander graph if $\forall X \subseteq L_i$ and $Y \subseteq L_j$ with $|X| > \gamma n'/d$ and $|Y| > \gamma n'/d$ there exists a directed edge $(x,y) \in E_m \cap X \times Y$.

Suppose that for each pair $i < j$ layers L_i and L_j are connected by a γ-bipartite expander graph with $\gamma < \frac{1}{5}$ and let the set $S \subseteq V_m$ contain $|S| \leq n'/10$ nodes from G_m. Call a layer L_i good if $|S \bigcap L_i| \leq \frac{n'}{5d}$. By Markov's inequality we have at least $g \geq d/2$ good layers. Now we claim that there is a path in $G_m - S$ traversing through every good layer (hence, $\mathsf{depth}(G_m - S) \geq d/2$). Let $Y_1, \dots, Y_g$ denote the good layers and let $R_1 = Y_1 - S$. We note that $|R_1| \geq \frac{4n'}{5d}$ by definition of a good layer. Once $R_1, \dots, R_i$ have been defined we let

$$R_{i+1} = \{v_{i+1} \in Y_{i+1} - S \ : \ \exists v_i \in R_i \cap \mathsf{ancestors}_{G_m - S}(Y_{i+1} - S)\}$$

denotes the set of meta-nodes in good layer Y_{i+1} that we can reach from R_i. A simple inductive argument shows that $|R_i| \geq \frac{3n'}{5d}$ for all $i \leq g$. Clearly, this holds for the base case since $|R_1| = |Y_1 - S| \geq \frac{4n'}{5d}$ by definition of a good layer. Suppose that $|R_i| \geq \frac{3n'}{5d}$ then, we claim that $|R_{i+1}| \geq \frac{3n'}{5d}$. To see this let $BAD_{i+1} = \{v_{i+1} \in Y_{i+1} \ : \ \mathsf{parents}(v_{i+1}) \cap R_i = \emptyset\}$ be the set of nodes in layer Y_{i+1} who do not have a parent in R_i. Because $|R_i| \geq \frac{3n'}{5d}$ and layers Y_i and Y_{i+1} are connected by a γ-bipartite expander graph ($\gamma < 1/5$) and we have $|BAD_{i+1}| \leq \gamma \frac{n'}{d} \leq \frac{n'}{5d}$. Finally, we have

$$|R_{i+1}| \geq \frac{n'}{d} - \left|S \bigcap Y_{i+1}\right| - |BAD_{i+1}| \geq \frac{n'}{d} - \left|S \bigcap Y_{i+1}\right| - \gamma\frac{n'}{d} \geq \frac{3n'}{5} \ ,$$

where the last inequality exploits the fact that $|S \bigcap Y_{i+1}| \leq \frac{n'}{5}$ by definition of a good layer.

Another simple inductive argument shows that, if $R_{i+1} \neq \emptyset$, then there must be a path of length $\geq i+1$ starting in R_1 and ending in R_{i+1}. Clearly, there is a path of length 1 to any node in R_1. Assume that for any nodes $x_1 \in R_1$ and $x_i \in R_i$ there is a path of length $\geq i$ starting at x_1 and ending x_i. Let $x_1 \in R_1$ and $x_{i+1} \in R_{i+1}$ be given. By definition there is some $x_i \in R_i$ such that $(x_i, x_{i+1}) \in E_m$. We can take a path of length $\geq i$ from x_1 to x_i and extend this path by x_{i+1}.

We now show that (with high probability) each fixed pair of layers L_i and L_j is connected with a γ-bipartite expander graph with $\gamma < 1/5$. [16] Recall that a pair of nodes $x, y \in V_m$ with $x < y$ are connected in G_m if there is an edge (u, v) in G with $u \in M_x^L$ (the last third of M_x) and $v \in M_y^F$ (the first third of M_y). Fix any pair of layers L_i and L_j $(i < j)$ and let $X \subset L_i$ and $Y \subset L_i$ be given. We have

$$\Pr[\forall x \in X, y \in Y \text{ we have } (x,y) \notin E_m] \leq \left(1 - \frac{|X|m}{3n}\right)^{|Y|m/3} ,$$

where the probability is taken over the sampling of the (n, δ, n)-random DAG G. To see this notice that, fixing $y \in L_j$ the event "$\forall x \in X, (x,y) \notin E_m$"' will occur if, when we sample the $m/3$ random edges into M_y^F, none of these edges 'hits' a node in the set $\bigcup_{x \in X} M_x^L$ and for each $v \in M_y^F$ we add an edge from $\bigcup_{x \in X} M_x^L$ to v with probability $\frac{|X|m}{3n}$. When $|X| = |Y| = \gamma n'/d$ we have

$$\Pr[\forall x \in X, y \in Y \text{ we have } (x,y) \notin E_m] \leq \left(1 - \frac{\gamma}{3d}\right)^{n\gamma/(3d)} \leq e^{\frac{-n\gamma^2}{9d^2}} .$$

Now we can union bound over all pairs of sets $X \subseteq L_i$ and $Y \subset L_i$ of size $\frac{\gamma n'}{d}$ to say that

$$\begin{aligned}\Pr[\exists X \subseteq L_i, Y \subseteq L_j \text{ s.t. } |X| = |Y| = \frac{\gamma n'}{d} \wedge E_m \cap X \times Y = \emptyset] &\leq \binom{\frac{n'}{d}}{\frac{\gamma n'}{d}}^2 e^{\frac{-n\gamma^2}{9d^2}} \\ &= \binom{\frac{n'}{d}}{\frac{\gamma n'}{d}}^2 e^{\frac{-100\gamma^2 n' \log n}{9d}}\end{aligned}$$

Union bounding over all $\binom{d}{2}$ pairs of layers $1 \leq i < j \leq d$ we can say that the probability that there exists a pair L_i, L_j of layers that is not connected by a γ-bipartite expander graph is at most

[16] Our analysis is similar to an argument of Erdös et al. [EGS75] demonstrating that a random bipartite graph with degree $\delta = \Omega(1)$ is a γ-bipartite expander with non-zero probability. In our setting we are *essentially* sampling a random bipartite graph with degree $\delta = \Omega(\log n)$. Thus, it is not surprising that the graph is a γ-bipartite expander graph with very high probability.

$$\binom{d}{2}\binom{\frac{n'}{d}}{\frac{\gamma n'}{d}}^2 e^{\frac{-100\gamma^2 n' \log n}{9d}} = o\left(e^{\left(3\gamma\ln\left(\frac{1}{\gamma}\right)+3(1-\gamma)\ln\left(\frac{1}{1-\gamma}\right)-\frac{100\gamma^2\log n}{9}\right)\left(\frac{n'}{d}\right)}\right) ,$$

where the last expression follows by Sterling's approximation. We note that the last term is negligibly small in n for any constant $\gamma > 0$ and any $n^{0.01} \le m \le n^{0.5}$.[17] □

From previous two claims it follows that G is $\left(\frac{n}{20m}, m^2/(600\log(n)), m\right)$-depth robust whenever $m \le \sqrt{n}$. Setting $m = n/(20e) < \sqrt{n}$ we obtain the desired result. □

Reminder of Theorem 5. *Let G be a (n, δ, n)-random DAG then, except with probability $o(n^{-7})$, we have*

$$\Pi^{\parallel}_{cc}(G) = \tilde{\Omega}\left(n^{5/3}\right) .$$

Proof of Theorem 5. Let G be a (n, δ, n)-random DAG, and let $G_1 = G - \{n/2+1, \ldots, n\}$ denote the subgraph on the first $n/2$ nodes. We observe that G_1 is itself a $(n/2, \delta, n/2)$-random DAG.

By setting $e = n^{2/3}$ in Lemma 2 we can find some fixed constant $c > 0$ such that our graph G_1 is $\left(n^{2/3}, cn^{2/3}/\log(n), n^{1/3}/20\right)$-block depth robust except with negligible probability in n. We observe that for any set $S \subseteq V$ of size $|S| \le n^{2/3}/2$ the graph $G_1 - N\left(S, n^{1/3}/20\right)$ is at least $(n^{2/3}/2, cn^{2/3}/\log(n))$-depth robust. Let $\tau > 0$ be a constant and consider a fixed segment $B_v = \{v, v+1, \ldots, v+200\tau n^{2/3}\log(n) - 1\} \subset \{n/2+1, \ldots, n\}$ of $\Omega\left(n^{2/3}\log(n)\right)$ consecutive nodes in the latter half of G. It will be useful to partition B_v into two sets $B_v^{first} = \{v, v+1, \ldots, v+100n^{2/3}\tau\log(n) - 1\}$ and $B_v^{last} = \{v + 100n^{2/3}\tau\log(n), \ldots, v+200n^{2/3}\tau\log(n) - 1\}$. We claim that, except with small probability, we will have $V(G_1) - N\left(S, n^{1/3}/20\right) \subseteq \mathsf{ancestors}_{G-S}(B_v^{last})$ for every set $S \subseteq V - B_v^{last}$ of size $|S| \le e/2$.

Claim 3 *Sample a (n, δ, n)-random DAG G. Then, except with probability $o\left(n^{-4.99\tau+2}\right)$, we will have $V(G_1) - N\left(S, \frac{n^{1/3}}{20}\right) \subseteq \mathsf{ancestors}_{G-S}\left(B_v^{last}\right)$ for every node $v \in [n - 200\tau n^{2/3}\log(n) - 1]$ and every set $S \subseteq V$ of size $|S| \le n^{2/3}/2$ s.t. $S \cap B_v^{last} = \emptyset$.*

Proof. We exploit the fact that G_1 is $\left(n^{2/3}, cn^{2/3}/\log(n), n^{1/3}/20\right)$-block depth robust. It will suffice to show, for every segment $L_u = \left\{u, \ldots, u + \frac{n^{1/3}}{20} - 1\right\}$ of $\frac{n^{1/3}}{20}$ consecutive nodes in G_1 and for every segment $B_v^{last} \subseteq \{n/2+1, \ldots, n\}$, that we have at least one directed edge (x, y) from some node $x \in L_u$ to a node $y \in B_v^{last}$ (except with small probability). Suppose that this holds, then for every

[17] Equivalently, for $n^{0.01}/(100\log n) \le d \le n^{0.5}/(100\log n)$). We also remark that we require $m \le \sqrt{n}$ so that the size of a layer $n'/d = 100n\log n/(m^2)$ is at least 1.

node $z \in S \cap \{1, \ldots, n/2\}$ we have an edge from some node in $N\left(z-1, \frac{n^{1/3}}{20}\right)$ to a node in B_v, which implies that $\mathsf{ancestors}_{G-S}\left(B_v^{last}\right) \supseteq V(G_1) - N\left(S, n^{1/3}/20\right)$ since G contains all of the edges $(i, i+1)$. Fixing v and u, the probability that no edge from L_u to B_v^{last} exists is at most

$$\left(1 - \frac{|L_u|}{n}\right)^{\left|B_v^{last}\right|} \leq \left(1 - \frac{1}{20n^{2/3}}\right)^{100\tau n^{2/3}\log(n)} \leq e^{-5\tau\log(n)} = n^{-5\tau} .$$

Union bounding over all $n^2/4$ pairs $(u, v) \in \{1, \ldots, n/2\} \times \{n/2+1, \ldots, n\}$ we obtain $n^{-5\tau+2} = o\left(n^{-4.99\tau+2}\right)$. $\square$

In the remainder of the proof we will assume that have $V(G_1) - N\left(S, \frac{n^{1/3}}{20}\right) \subseteq \mathsf{ancestors}_{G-S}(B_v^{last})$ for every node $v \in [n - 200\tau n^{2/3}\log(n) - 1]$ and every set $S \subseteq V$ of size $|S| \leq n^{2/3}/2$ s.t. $S \cap B_v^{last} = \emptyset$. We can set $\tau = 2$ to ensure that the probability this happens is $o\left(n^{-7}\right)$.

Now consider any fixed segment $B_v = \{v, v+1, \ldots, v + 200\tau n^{2/3}\log(n) - 1\} \subset \{n/2+1, \ldots, n\}$ of nodes. That is, B_v is a segment of $200\tau n^{2/3}\log(n)$ consecutive nodes in the second half of G. In the following claim we lowerbound the cost incurred while pebbling B_v. Let t_1 (resp. t_2, t_3) denote the first time at which we place a pebble on node v (resp. node $v + 100n^{2/3}\tau\log(n) - 1$, node $v + 200n^{2/3}\tau\log(n) - 1$). Intuitively, t_1 denotes the first time step in which we place a pebble on the set B_v^{first}, t_2 denotes the time step at which we finish pebbling B_v^{first} and t_3 denotes the time step at which we finish pebbling B_v^{last}.

Claim 4 *Let $P_0, \ldots, P_t$ denote a pebbling of G then*

$$\sum_{i \in [t_1, t_3]} |P_i| = \tilde{\Omega}\left(n^{4/3}\right) .$$

Proof. There are two cases:

1. For every $i \in [t_1, t_2]$ we have $|P_i| \geq n^{2/3}/2$ pebbles on G. Total Cost: $\sum_{i \in [t_1, t_2]} |P_i| \geq n^{2/3}(t_2 + 1 - t_1)/2 = \Omega\left(n^{4/3}\right)$.
2. For some $t' \in [t_1, t_2]$ we have at most $|P_{t'}| < n^{2/3}/2$ pebbles on G.

In the second case we set $S = P_{t'}$ and we note that by the previous claim we have $V(G_1) - N\left(S, \frac{n^{1/3}}{20}\right) \subseteq \mathsf{ancestors}_{G-S}\left(B_v^{last}\right)$. We we also recall that the graph $G_1 - N\left(S, n^{1/3}/20\right)$ is at least $(n^{2/3}/2, cn^{2/3}/\log(n))$-depth robust for some constant c. By Corollary 2 we have $\sum_{t \in [t_2, t_3]} |P_i| \geq \Pi_{cc}^{\parallel}\left(G - S, B_v^{last}\right) \geq \frac{cn^{4/3}}{2\log n} = \Omega\left(\frac{n^{4/3}}{\log n}\right)$ which concludes the proof of the claim. $\square$

To complete the proof of Theorem 5 it remains only to observe that the previous claim implies that the total cost of a pebbling is at least $\frac{n}{2|B_v|}\tilde{\Omega}\left(n^{4/3}\right) = \tilde{\Omega}\left(n^{5/3}\right)$. $\square$

Reminder of Theorem 6.

$$G \in \mathbb{D}_{\epsilon,g}^{\lambda,k} \quad \Rightarrow \quad \Pi_{cc}^{\parallel}(G) \geq \epsilon\lambda g\left(\frac{k}{2}-g\right).$$

Proof of Theorem 6. We number the nodes of $G = (V, E)$ in a topological order with the set $[n]$. Let $\{L_i \subseteq V\}$ be the λ disjunct node subsets described in Definition 7. For any L_i and $j \in [\lfloor k/2g \rfloor]$ let $L_{i,j}$ be the j^{th} interval of $2g$ consecutive nodes in L_i. That is if $L_i = [a, a+k-1]$ then $L_{i,j} = [a+i2g, a+(i+1)2g-1)]$. Let G_i denote the subgraph of G consisting of the nodes up to the end of L_i. That is G consists exactly of the node set $[a+(i+1)2g-1]$ and edges of G between those nodes.

Let $P = (P_1, P_2, \ldots)$ be a legal pebbling of G and let $\bar{t}_v$ denote the first time step at which node v is pebbled by P. We use the following shorthand:

$$c_{i,j} := \sum_{i=\bar{t}_a}^{\bar{t}_z} |P_i| \qquad\qquad c_i := \sum_{i=\bar{t}_\alpha}^{\bar{t}_\omega} |P_i|$$

where a and z are the first and last nodes of $L_{i,j}$ and α and ω are the first and last nodes of L_i respectively. In particular $\mathsf{cc}(P) \geq \sum_{i\in[\lambda]} c_j$ and our goal is to lowerbound this sum. The theorem follows from the next claim.

Claim 5 *For all $i \in [\lambda]$ and $j \in [\lfloor k/2g \rfloor]$ we have $c_{i,j} \geq \epsilon g^2$.*

Proof. Fix any i and j as in the claim. Let L and R be the first and second halves of $L_{i,j}$ and let a and y be the first and last nodes of L respectively and let z denote the last node of R. As G_i is (ϵ, g)-dispersed there exists a set Γ of ϵg node disjunct paths each of length g, terminating R and with no other nodes in L_i. Let $p \in \Gamma$ be such a path.

Now either p contains (at least) one pebble during all time steps in $[\bar{t}_a, \bar{t}_y]$ or not. If so p contributes at least g to $c_{i,j}$ with pebbles not lying on any of other $p' \in \Gamma$. If not then in order to pebble z, the last node of R, all of p must also be pebbled between steps $[\bar{t}_a, \bar{t}_z]$. Thus again p contributes at least g to $c_{i,j}$ with pebbles not lying on any other $p' \in \Gamma$.

Summing over all paths in Γ we get that $c_{i,j} \geq \epsilon g^2$ which concludes the proof of the claim and theorem. $\square$

In particular, from Claim 5, it immediately follows that

$$c_i \geq \left\lfloor \frac{k}{2g} \right\rfloor (\epsilon g^2) \geq \epsilon g\left(\frac{k}{2}-g\right).$$

Therefore it follows that

$$\Pi_{cc}^{\parallel}(G) \geq \sum_{i\in[\lambda]} c_i \geq \epsilon\lambda g\left(\frac{k}{2}-g\right).$$

□

Reminder of Lemma 8. *Let $G = (V, E)$ be a DAG of size n, indegree δ and* $\mathsf{depth}(G) \le d_0$*. If G is (e_1, d_1)-reducible with parameters e_1, d_1 such that $2d_1 n \le e_1 d_0$ and $d_1 \le d_0$ then for any target set $T \subseteq V$ we have*

$$\Pi^{\parallel}_{cc}(G, T, 2d_0) \le (4\delta + 4)\, e_1 d_0 + \frac{n}{e_1}\left(\max_{\substack{T' \subseteq V - S_1 \\ |T'| \le \delta \cdot e_1}} \Pi^{\parallel}_{cc}\left(G - S_1, T', 2d_1\right)\right) ,$$

where $S_1 \subseteq V$ has size $|S_1| \le e_1$ such that $\mathsf{depth}\left(G - S_1\right) \le d_1$*.*

Proof. Let $G = (V, E), S_1, T \subseteq V$ be given such that $\mathsf{depth}(G) \le d_0$, $|V| = n$ and $\mathsf{depth}(G - S_1) \le d_1$ and $2d_1 n \le e_1 d_0$ where $e_1 \ge |S_1|$. We begin by constructing a partition $D_1, \ldots, D_{2d_1}$ of V which satisfies the following properties:

SOURCES: $\mathsf{parents}\big(D_1\big) = \emptyset$,
TOPOLOGICALLY ORDERED: $\forall i \le 2d_0 - 1 \quad \mathsf{parents}\big(D_{i+1}\big) \subseteq \bigcup_{j \le i} D_j$, and
MAXIMUM SIZE: $\forall i \le 2d_0 \quad |D_i| \le \frac{n}{d_0}$.

To do this we can simply partition the nodes V according to their depth and then further split the sets which are larger than n/d_0. We let $U_i = \cup_{j=1}^{i} D_j$ denote the union of all sets D_j with $j < i$. In our pebbling $P_1, \ldots, P_{2d_0}$ we will maintain the invariants that $P_i \supset D_i$ for all $i \le 2d_0$ and that $P_i \supset S_1 \cap U_i$. The first invariant ensures that we finish in $2d_0$ steps and the second ensures that we never discard pebbles from the set S_1 to that the induced DAG $G - S_1 - (V - U_i)$ has depth at most d_1.

We let $m = e_1/(n/d_0) \ge 2d_1$ and divide pebbling rounds into blocks of m: $P_1, \ldots, P_m, P_{m+1}, \ldots, P_{2m}, \ldots$. We call each group $P_{jm+1}, \ldots, P_{jm+m}$ a light phase. Given $k = jm + i \le 2d_0$ with $0 \le i < m$ we let $X_k = \mathsf{parents}(U_{jm+m} \backslash U_{k+1}) \cap U_k$ denote the parents of nodes that we still plan to pebble for the first time in the current light phase excluding parents that have not yet been pebbled themselves. It will be useful to partition X_k into two sets $X_k^- = (X_k \cap U_{jm-2d_1}) \backslash S_1$ and $X^+ = X_k \setminus X_k^-$. We will maintain the invariant that $P_k \supset X_k$. To maintain this invariant we will occasionally need to execute a balloon phase during which we recover previously discarded pebbles.

We observe that, by definition, for each $j \le 2d_0/m$ we can find a pebbling $Q_1^j, \ldots, Q_{2d_1}^j$ of $G - S_1 - (V - U_{jm-2d_1})$ with target set $X_{jm-2d_1}^-$ and cost at most

$$\sum_{i=1}^{2d_1} \left|Q_i^j\right| \le \max_{\substack{T' \subseteq V - S_1 \\ |T'| \le \delta \cdot e_1}} \Pi^{\parallel}_{cc}\left(G - S_1, T', 2d_1\right) .$$

For the j^{th} balloon phase we will paste $Q_1^j, \ldots, Q_{2d_1}^j$ onto the pebbling rounds $P_{jm-2d_1+1}, \ldots, P_{jm}$ to ensure that $P_{jm-2d_1+i} \supset Q_i^j$ for each $1 \le i \le 2d_1$. We now define our pebbling attack formally. During a balloon phase $k = jm - 2d_1 + i$ with $1 \le i \le 2d_1$ we set

$$P_k = D_k \cup ((S_1 \cup X_k \cup X_{jm}) \cap P_{k-1}) \cup Q_i^j ,$$

and during a light phase $k = jm + i$ with $1 \leq i \leq m - 2d_1 - 1$ we set

$$P_k = D_k \cup (S_1 \cap P_{k-1}) \cup (X_k \cap P_{k-1})$$

We now show that this is a legal pebbling. To show this it will help to establish three invariants which we do in the following three claims. Intuitively, the first claim states that we always maintain pebbles on S_1. The second claim states that we start each light phase with pebbles on all of the necessary parents for that light phase. The third claim states that we maintain the necessary pebbles on all of the necessary parents throughout the light phase.

Claim 6 *For each $1 \leq i \leq 2d_0$ we have $P_i \supset (S_1 \cap U_i)$.*

Proof. We use induction. For the base case we have $P_1 = D_1 = U_1 \supset (S_1 \cap U_1)$. Assuming that $P_i \supset (S_1 \cap U_i)$ some $i \geq 1$ we apply the definition of P_i to see that

$$\begin{aligned} P_{i+1} &\supset D_{i+1} \cup (S_1 \cap P_i) \\ &\supset D_{i+1} \cup (S_1 \cap U_i) \\ &\supset S_1 \cap U_{i+1} \ . \end{aligned}$$

□

Claim 7 *For each $1 < j \leq 2d_0/m$ we have $P_{jm} \supset X_{jm}$.*

Proof. Let j be given. We argue inductively that for each $i \leq 2d_1$ we have

$$P_{jm-2d_1+i} \supset \left(X_{jm}^{-} \cap \bigcup_{r=1}^{i} Q_r^j \right) \cup \left(\bigcup_{r=1}^{i} D_{jm-2d_1+r} \cap X_{jm}^{+} \right) \ . \qquad (2)$$

Trivially, equation 2 holds at step $i = 1$ since, by the definition of P_i, when we set $k = jm - 2d_1 + 1$ we have

$$P_k \supset Q_1^j \cup D_k \supset \left(X_{jm}^{-} \cap Q_1^j \right) \cup \left(X_{jm}^{+} \cap D_k \right) \ .$$

Now assume that equation 2 holds for some $2d_1 > i \geq 1$ then by the definition of P_i

$$\begin{aligned} P_{jm-2d_1+i+1} &\supset D_{jm-2d_1+i+1} \cup Q_{i+1}^j \cup (X_{jm} \cap P_{jm-2d_1+i}) \\ &\supset D_{jm-2d_1+i+1} \cup Q_{i+1}^j \cup \left(X_{jm}^{-} \cap \bigcup_{r=1}^{i} Q_r^j \right) \cup \left(\bigcup_{r=1}^{i} D_{jm} \cap X_{jm}^{+} \right) \\ &\supset \left(X_{jm}^{-} \cap \bigcup_{r=1}^{i+1} Q_i^j \right) \cup \left(\bigcup_{r=1}^{i+1} D_{jm} \cap X_{jm-2d_1+i}^{+} \right) \ . \end{aligned}$$

Setting $i = 2d_1$ we have $P_{jm} \supset \left(X_{jm}^{-} \cap \bigcup_{r=1}^{2d_1} Q_r^j \right) \supset X_{jm}^{-}$, since $Q_1^j, \ldots, Q_{2d_1}^j$ has target set X_{jm}^{-}. We also have $P_{jm} \supset X_{jm}^{+} \cap \bigcup_{r=1}^{2d_1} D_{jm-2d_1+r} \supset X_{jm}^{+} \setminus S_1$. Finally, since $P_{jm} \supset S_1 \cap U_{jm}$ by Claim 6, we have $P_{jm} \supset X_{jm}^{+} \cup X_{jm}^{-} = X_{jm}$.
□

Claim 8 *For each $k > 0$ we have $P_k \supset X_k$.*

Proof. Let $k = jm+i$ with $0 \leq i < m$. We argue by induction on i. For the base case we note that if $i = 0$ then we already have $P_k = P_{jm} \supset X_{jm}$ by the previous claim. Now assume that we have $P_{jm+i} \supset X_{jm+i}$ for some $0 \leq i < m-1$ then

$$\begin{aligned}
P_{jm+i+1} &\supset D_{jm+i+1} \cup (X_{jm+i+1} \cap P_{jm+i}) \\
&\supset D_{jm+i+1} \cup (X_{jm+i+1} \cap X_{jm+i}) \\
&= D_{jm+i+1} \cup ((\mathsf{parents}\,(U_{jm+m} \backslash U_{jm+i+2}) \cap U_{jm+i+1}) \cap (\mathsf{parents}\,(U_{jm+m} \backslash U_{jm+i+1}) \cap U_{jm+i})) \\
&= D_{jm+i+1} \cup (\mathsf{parents}\,(U_{jm+m} \backslash U_{jm+i+2}) \cap U_{jm+i}) \\
&\supset \mathsf{parents}\,(U_{jm+m} \backslash U_{jm+i+2}) \cap U_{jm+i+1} \\
&= X_{jm+i+1}
\end{aligned}$$

□

Let $k \geq 0$ be given. We now argue that pebbling step P_{k+1} is legal. Clearly, P_1 is a legal pebbling step since $P_1 = D_1 \subseteq \{v : \mathsf{indeg}(v) = 0\}$. Suppose that $k > 1$. There are two cases:

Case 1: Balloon Phase. $k = jm - 2d_1 + i$ for $0 \leq i < 2d_1$. In this case we note that, by construction, $P_{k+1} \backslash P_k \subseteq D_{k+1} \cup Q_{i+1}^j$ for all $0 \leq i < m$. Thus, the pebbling step is legal if $\mathsf{parents}\left(Q_{i+1}^j \cup D_{k+1}\right) \subseteq P_k$. Since $Q_1^j, \ldots, Q_{2d_1}^j$ is a legal pebbling of $G - S_1 - (V - U_{jm-2d_1})$ we must have $\mathsf{parents}\left(Q_{i+1}^j\right) \subseteq Q_i^j \cup (S_1 \cap U_{jm-2d_1}) \subseteq P_k$ by the first claim. Note that we define $Q_0^j = \emptyset$ for convenience when $i = 0$. Similarly, we have $\mathsf{parents}\,(D_{k+1}) \subseteq X_k \subseteq P_k$ by the third claim and the definition of X_k.

Case 2: Light Phase. $k = jm + i$ for some $0 \leq i < m - 2d_1$. In this case we note that, by construction $P_{k+1} \backslash P_k \subseteq D_{k+1}$ for all $1 \leq i \leq m - 2d_1$. To show that this pebbling step is legal we observe that $\mathsf{parents}\,(D_{k+1}) \subseteq X_k \subseteq P_k$ by the third claim and the definition of X_k.

This completes the proof that the pebbling is legal. We now analyse the pebbling cost. We observe that $|X_{jm+i}| \leq |\mathsf{parents}(U_{jm+m} \backslash U_{jm+1})| \leq \delta \cdot m \cdot \frac{n}{d_0} = \delta \cdot e_1$.

We have

$$\begin{aligned}
\sum_{i=1}^{2d_0} |P_i| &\leq \sum_{i=1}^{2d_0} \left(|D_i| + |S_1 \cap P_{i-1}| + |X_i \cap P_{i-1}|\right) \\
&\quad + \sum_{j=0}^{d_0/m} \sum_{i=1}^{2d_1} \left(\left|Q_i^j\right| + \left|X_{jm} \cap P_{jm-2d_1+i-1}\right|\right) \\
&\leq \left(\sum_{i=1}^{2d_0} |D_i|\right) + 2d_0e_1 + 2d_0\left(\delta mn/d_0\right) \\
&\quad + 2d_0\left(\delta mn/d_0\right) + \sum_{j=0}^{n/e_1} \sum_{i=1}^{2d_1} \left|Q_i^j\right| \\
&= n + 2d_0e_1(1+2\delta) + \left(\frac{n}{e_1} + 1\right)\left(\max_{\substack{T' \subseteq V - S_1 \\ |T'| \leq \delta \cdot e_1}} \Pi_{cc}^{\parallel}\left(G - S_1, T', 2d_1\right)\right) \\
&\leq n + 2d_0e_1(1+2\delta) + 2d_1n + \frac{n}{e_1}\left(\max_{\substack{T' \subseteq V - S_1 \\ |T'| \leq \delta \cdot e_1}} \Pi_{cc}^{\parallel}\left(G - S_1, T', 2d_1\right)\right) \\
&\leq d_0e_1(2+4\delta) + (d_0e_1 + n) + \frac{n}{e_1}\left(\max_{\substack{T' \subseteq V - S_1 \\ |T'| \leq \delta \cdot e_1}} \Pi_{cc}^{\parallel}\left(G - S_1, T', 2d_1\right)\right) \\
&\leq d_0e_1(4+4\delta) + \frac{n}{e_1}\left(\max_{\substack{T' \subseteq V - S_1 \\ |T'| \leq \delta \cdot e_1}} \Pi_{cc}^{\parallel}\left(G - S_1, T', 2d_1\right)\right) .
\end{aligned}$$

□

Reminder of Theorem 8. *Let G be a f-reducible DAG on n nodes then if $f(d) = \tilde{O}\left(\frac{n}{d^b}\right)$ for some constant $0 < b \leq 2/3$ and let $a = \frac{1-2b+\sqrt{1+4b^2}}{2}$. Then for any constant $\epsilon > 0$*

$$\Pi_{cc}^{\parallel}(G) \leq O\left(n^{1+a+\epsilon}\right) .$$

Proof. As usual let $\delta = \mathsf{indeg}(G)$. As long as $b \leq 2/3$ it follows that $a \geq b$ so we can find a monotonic sequence $a_1 = a \leq a_2 \leq \ldots \leq 1$ with the following properties: (1) $a_n \to 1$ as $n \to \infty$, (2) $1 + \frac{a_i - a_{i+1}}{b} - a_{i+1} \leq 0$ for all $i \geq 1$, (3) $a_{j+1} + \frac{1-a_j}{b} + j - \sum_{i=1}^{j} a_i = 1 + a$ for all $j > 1$, and (4) $j - \sum_{i=1}^{j} a_i \to a$ as $j \to \infty$. In particular, we claim that, if $b \leq a$, the sequence

$$a_{i+1} = 1 + \frac{(a-1)(1-a_i)}{b}, \text{ where } \qquad a_1 = a = \frac{1-2b+\sqrt{1+4b^2}}{2}$$

satisfies all four required properties. We remark that a is the positive solution to the equation $a^2 + (2b-1)a - b = 0$.

Assuming for now that such a sequence exists we can define a sequence $e_1 = n^{a_1+\epsilon/3}, e_2 = n^{a_2+\epsilon/3}, \ldots$. Because G is f reducible with $f(d) = \tilde{O}\left(\frac{n}{d^b}\right)$ we can find depth-reducing sets $S_0 = \emptyset, S_1, S_2, \ldots \subseteq V$ s.t $|S_i| \le e_i$ and $d_i = \mathsf{depth}(G - S_i) \le n^{\frac{1-a_i}{b}}$ for each $i > 0$. Applying property (2) we observe that for each $i > 1$ we have

$$d_{i+1}n = n^{1+\frac{1-a_{i+1}}{b}} \le n^{a_{i+1}+\frac{1-a_i}{b}} \le e_{i+1}d_i/2 \ .$$

Thus, we satisfy the conditions necessary to invoke Lemma 8 recursively. To simplify notation in the remainder of the proof we will let

$$R_i \doteq \max_{\substack{T \subseteq V - S_i \\ |T| \le \delta e_i}} \Pi_{cc}^{\parallel}\left(G, T, 2d_i\right) \ .$$

Property (1) ensures that for any constant ϵ we can find a constant k such that $a_k \ge 1 - \epsilon b/2$ and $k - \sum_{i=1}^{k} a_i \le a + \epsilon/3$. Hence, $d_k = n^{(1-a_k)/b} \le n^{\epsilon/3}$. We use property (3) to unroll the recurrence using Lemma 8. In particular, an inductive argument shows that when we unroll j times using Lemma 8 we get

$$\Pi_{cc}^{\parallel}\left(G, \{n\}, 2d_0\right) \le j\left(4\delta + 4\right) n^{1+a+\epsilon/3} + n^{j-\sum_{i=1}^{j} a_i}\left(R_j\right) \ . \qquad (3)$$

Setting $j = k$ and applying property (4) to equation 3 we now have $\Pi_{cc}^{\parallel}(G) \le$

$$\begin{aligned}
\Pi_{cc}^{\parallel}\left(G, \{n\}, 2d_0\right) &\le k\left(4\delta + 4\right) n^{1+a+\epsilon/2} + n^{k-\sum_{i=1}^{k} a_i}\left(R_k\right) \\
&\le k\left(4\delta + 4\right) n^{1+a+\epsilon/3} + n^{k-\sum_{i=1}^{k} a_i}\left(nd_k\right) \\
&\le k\left(4\delta + 4\right) n^{1+a+\epsilon/3} + n^{a+\epsilon/3}\left(n^{1+\epsilon/3}\right) \\
&\le k\left(4\delta + 4\right) n^{1+a+\epsilon/3} + n^{1+a+2\epsilon/3} \\
&= o\left(n^{1+a+\epsilon}\right) \ .
\end{aligned}$$

It remains to show that equation 3 holds for each $j \ge 1$. For the base case $j = 1$ we apply Lemma 8 once with e_1, d_1 to get

$$\Pi_{cc}^{\parallel}\left(G, \{n\}, 2d_0, n\right) \le (4\delta{+}4)e_1d_0 + \frac{n}{e_1}\left(R_1\right) = (4\delta{+}4)n^{1+a+\epsilon/3} + n^{1-a_1-\epsilon/3}\left(R_1\right) \ .$$

For the inductive step assume that for some $j \ge 1$ we have

$$\Pi_{cc}^{\parallel}\left(G, \{n\}, 2d_0, n\right) \le j(4\delta + 2)n^{1+a+\epsilon/3} + n^{j-\sum_{i=1}^{j} a_i}\left(R_j\right) \ .$$

Then we can apply Lemma 8 to claim that

$$R_j \le (4\delta{+}4)e_{j+1}d_j + \frac{n}{e_{j+1}}\left(R_{j+1}\right) = (4\delta{+}2)n^{a_{j+1}+\frac{1-a_j}{b}+\epsilon/3} + n^{1-a_{j+1}-\epsilon/3}\left(R_{j+1}\right) \ .$$

Plugging into equation 3 and applying properties (3) and (4) we get

$$\begin{aligned}
\Pi_{cc}^{\parallel}\left(G, \{n\}, 2d_0, n\right) &\le (4\delta + 4)\left(jn^{1+a+\epsilon/3} + n^{a_{j+1}+\frac{1-a_j}{b}+\epsilon/3+j-\sum_{i=1}^{j} a_i}\right) \\
&\quad + n^{j+1-\sum_{i=1}^{j+1} a_i-\epsilon/3}\left(R_{j+1}\right) \\
&\le (4\delta + 4)\left(jn^{1+a+\epsilon/3} + n^{1+a+\epsilon/3}\right) + n^{j+1-\sum_{i=1}^{j+1} a_i-\epsilon/3}\left(R_{j+1}\right) \ .
\end{aligned}$$

It remains to show that our sequence $a_1 \le a_2 \le \ldots$ satisfies all four required properties. This is established in the next three claims. The first describes a helper function. The second establishes properties (2) and (3) and the third claim establishes properties (1) and (4).

Claim 9 *For $i \ge 1$ we have*

$$0 \le a_i \le 1, \qquad a_{i+1} \ge a_i, \quad \textit{and} \qquad a_{i+1} + \frac{1-a_i}{b} = 1 - a_{i+1} + a_{i+2} + \frac{1-a_{i+1}}{b} \ .$$

Proof. We first two statements by induction. Clearly, $0 \le a = a_1 \le 1$. Now for $i \ge 1$ we have

$$a_{i+1} = 1 + \frac{(a-1)(1-a_i)}{b} = 1 - \left(\frac{(1-a)}{b}\right)(1-a_i) \le 1 \ ,$$

because b, $(1-a)$ and $(1-a_i)$ are all positive values. Similarly, we also have

$$a_{i+1} - a_i = 1 - \left(\frac{(1-a)}{b}\right)(1-a_i) - a_i = (1-a_i)\left(1 + \frac{(1-a)}{b}\right) \ge 0 \ .$$

Thus, $a_{i+1} \ge a_i \ge 0$.

For the third statement we observe that $a_{i+1} + \frac{1-a_i}{b} = \frac{a \cdot a_{i+1} - 1}{a-1}$ and that $1 - a_{i+1} + a_{i+2} + \frac{1-a_{i+1}}{b} = 2 - a_{i+1} + \frac{a(1-a_{i+1})}{b}$. Now we claim that $2 - a_{i+1} + \frac{a(1-a_{i+1})}{b} - \frac{a \cdot a_{i+1} - 1}{a-1} = 0$. Multiplying both sides by $(a-1)b$ we get $2(a-1)b - a_{i+1}(a-1)b + a(a-1)(1-a_{i+1}) - ab \cdot a_{i+1} + b = 0$. Refactoring we get $-b(1-a_{i+1}) + a^2(1-a_{i+1}) + a((2b-1)(1-a_{i+1})) = 0$. Dividing by $(1-a_{i+1})$ we get $a^2 + (2b-1)a - b = 0$, which is true by definition of a. $\square$

Claim 10 $1 + \frac{a_i - a_{i+1}}{b} - a_{i+1} \le 0$ *for all $i \ge 1$ and* $a_{j+1} + \frac{1-a_j}{b} + j - \sum_{i=1}^{j} a_i = 1 + a$ *for all $j > 1$.*

Proof. We first note that

$$\begin{aligned}
1 + \frac{a_i - a_{i+1}}{b} - a_{i+1} &= \frac{a_i - a_{i+1} - (a-1)(1-a_i)}{b} \\
&= \frac{-\left(1 + \frac{(a-1)(1-a_i)}{b}\right) + a(a_i - 1) + 1}{b} \\
&= \frac{(a-1+ab)(a_i-1) + b - 1}{b^2} \\
&\le \frac{(a-1+ab)(a_i-1) + a - 1}{b^2} \\
&\le \frac{(a-1)a_i + ab(a_i-1)}{b^2} \\
&\le 0 \ .
\end{aligned}$$

where the last inequality follows because $a, a_i < 1$ and $a, b, a_i > 0$. In the first inequality we exploited the fact that $a \ge b$. Now we show that $a_{j+1} + \frac{1-a_j}{b} +$

$j - \sum_{i=1}^{j} a_i = 1 + a$ by induction. For the base case ($j = 2$) we have

$$\begin{aligned}
a_3 + \frac{1-a_2}{b} + 2 - \sum_{i=1}^{2} a_i &= (1 - a_1) + \left(1 - a_2 + a_3 + \frac{1-a_2}{b}\right) \\
&= (1 - a_1) + \left(a_2 + \frac{1-a}{b}\right) \\
&= (1 - a_1) + \left(1 + \frac{(a-1)(1-a)}{b} + \frac{1-a}{b}\right) \\
&= 1 + \frac{b(1-a) + (a)(1-a)}{b} \\
&= 1 - \frac{-b + a(b-1) + ab - ab + a^2)}{b} \\
&= 1 - \frac{-b + a(2b-1) + a^2)}{b} + a \\
&= 1 + a \ .
\end{aligned}$$

Where the first line follows by Claim 9 and the last line exploits the quadratic relationship $-b + a(2b-1) + a^2 = 0$. Now by induction and by the first claim we have

$$\begin{aligned}
1 + a &= a_{j+1} + \frac{1-a_j}{b} + j - \sum_{i=1}^{j} a_i \\
&= \left(1 - a_{j+1} + a_{j+2} + \frac{1-a_{j+1}}{b}\right) + j - \sum_{i=1}^{j} a_i = 1 + a \\
&= a_{j+2} + \frac{1-a_{j+1}}{b} + (j+1) - \sum_{i=1}^{j+1} a_i
\end{aligned}$$

□

Claim 11

$$\lim_{i\to\infty} a_i = 1, \quad \textit{and} \qquad \lim_{i\to\infty} \sum_{j=1}^{i} (1 - a_i) = a \ .$$

Proof. We first note that $0 < \left(\frac{1-a}{b}\right) < 1$, and that

$$\begin{aligned}
1 - a_{i+1} &= \frac{(1-a)(1-a_i)}{b} \\
&= \left(\frac{(1-a)}{b}\right)(1 - a_i) \\
&= \left(\frac{(1-a)}{b}\right)^2 (1 - a_{i-1}) \\
&= \dots \\
&= \left(\frac{(1-a)}{b}\right)^i (1 - a_1) \ .
\end{aligned}$$

Therefore, $\lim_{i\to\infty} a_i = 1 - \lim_{i\to\infty}\left(\frac{(1-a)}{b}\right)^i (1-a_1) = 1$. Similarly,

$$\sum_{j=1}^{i}(1-a_j) = \sum_{j=0}^{i-1}\left(\frac{(1-a)}{b}\right)^j (1-a_1)$$
$$= (1-a)\sum_{j=0}^{i-1}\left(\frac{(1-a)}{b}\right)^j .$$

Thus, $\lim_{i\to\infty}\sum_{j=1}^{i}(1-a_j) = \frac{1-a}{1-\frac{1-a}{b}} = \frac{b-ba}{b-1+a} = a$, where the last equality follows because a was chosen so that $a^2 + (2b-1)a - b = 0$. □

□

Reminder of Lemma 9. *Let $f_b(d) = \tilde{O}\left(\frac{n}{d^b}\right)$ then*

1. *Let $\delta = O(\mathsf{polylog}(n))$ then a (n,δ,n)-random DAG is $f_{0.5}$-reducible with high probability.*
2. *The Catena DAGs DFG^n_λ and BFG^n_λ are both f_1-reducible for $\lambda = O(\mathsf{polylog}(n))$.*
3. *The Balloon Hashing Linear (and the DB) graph Lin^σ_τ is f_1-reducible for $\tau = O(\mathsf{polylog}(n))$.*

The proof of Lemma 9 closely follows arguments from Alwen and Blocki [AB16], who proved these DAGs were (e,d)-reducible for specific values of e and d. Because the attack of Alwen and Blocki [AB16] was non-recursive they only focused on proving (e,d)-reducible for the values e,d which optimized the quality of their attack.

Proof of Lemma 9. (sketch) We first consider an arbitrary $\lambda = O(\mathsf{polylog}(n))$-layered DAG G. This includes Catena DAGs DFG^n_λ and BFG^n_λ as well as Balloon Hashing Linear (and the Double Buffer) graph Lin^σ_τ (with $\tau = \lambda = O(\mathsf{polylog}(n))$). Let d be given and let $e = (\lambda+1)n/d$. For simplicity assume that e, $n/(\lambda+1)$ and $d/(\lambda+1)$ are integers[18]. Let $S = \{i \times d/(\lambda+1) : i \le g\}$ and observe that S has size $|S| = e = \tilde{O}(n/d)$. Thus, to show that G is f_1 reducible it suffices to show that $\mathsf{depth}(G - S) \le d$. Define $L_i = \{i \times n/(\lambda+1)+1, \ldots, (i+1)\times n/(\lambda+1)\}$ for $0 \le i \le \lambda$. We note that any path in $G - S$ can remain on layer L_i for at most $d/(\lambda+1)$ steps before moving to a higher layer and there are $\lambda+1$ layers. Thus, the maximum length of any path is $(\lambda+1)d/(\lambda+1) = d$.

Next we consider with a (n,δ,n)-random DAG G on nodes $\{1,\ldots,n\}$. Let d be given and let $g = n/\sqrt{d}$. For simplicity assume that $\sqrt{d}$ and g are integers[19]. Let $S_1 = \{i \times \sqrt{d} : i \le g\}$ and observe that S_1 has size $|S_1| = g$. Define $L_i = \{i\times g+1, \ldots, i\times g+g\}$. We call an node $v \in L_i$ good if $\mathsf{parents}(v)\bigcap L_i = \emptyset$ and we let $B_i = \{v \in L_i : \mathsf{parents}(v)\bigcap L_i \neq \emptyset\}$ denote the set of all bad nodes in L_i. Finally, we let $S = S_1 \cup \bigcup_{i=0}^{\sqrt{d}-1} B_i$. It is easy to verify that $\mathsf{depth}(G-S) \le d$

[18] This allows us to simply presentation by ignoring insignificant rounding terms.
[19] This allows us to simply presentation by ignoring insignificant rounding terms.

because any path in $G-S$ can remain on layer L_i for at most $\sqrt{d}$ steps before moving up to a higher layer and there are $\sqrt{d}$ layers. Thus, the maximum length of any path is $\sqrt{d}^2 = d$. It is easy to see that $\mathbb{E}\left[|B_i|\right] \leq \frac{|B_i|}{i+1} = \frac{\delta g}{i+1}$ — let x_v denote the indicator random variable for the event $v \in B_i$ then $\Pr[x_v = 1] \leq \frac{1}{i+1}$. Thus, $\mathbb{E}\left[|S|\right] \leq g + \delta g \sum_{i=1} i^{-1} = O\left(\frac{\delta n \log n}{\sqrt{d}}\right)$. Furthermore, standard concentration bounds imply that $|S| - 2\mathbb{E}\left[|S|\right] \leq 0$ except with very small probability. Thus, a (n, δ, n)-random DAG G is $f_{0.5}$-reducible with high probability. □

B.1 Tighter Bounds on Memory-Hardness

As an example of the power of our results we can immediately improve on the analysis of the high CC graph of [AS15]. In that work, first a graph $G_{n,\delta}$ of size n and indegree $\delta \leq \log^3(n)$ is given and it is show to have CC approximately $n^2/\log(n)$. Next the indegree is reduced to obtain $G'_{n*\delta,2}$ at a cost of δ^3 to the CC. That is G' has size $N = n*\delta$ and CC of roughly $N^2/\log^{10}(N)$.

We can immediately improve on this result using our new indegree reduction. In particular Lemma 1 only loses a factor of δ in the CC when reducing the indegree. Moreover, in [AS15] $G_{n,\delta}$ inherits its indegree directly from the depth-robust graphs of [MMV13] which are estimated (for simplicity) by [AS15] to have indegree $\log^3(n)$ when in fact they have indegree $\log^2(n)$ times some polyloglog function of n. So the same is true for $G_{n,\delta}$. Ignoring polyloglog factors and setting $g = n/\log^3(n)$ in Theorem 10 we see that G must be at least (e,d)-depth-robust for some $e = \Omega(n/\log(n))$ and $d = \Omega(n/\log^4(n))$. So we can reduce the indegree of $G_{n,\delta}$ to 2 using Lemma 1 with $\gamma = \delta$ and we obtain a graph $H \in \mathbb{G}_{2n\delta,2}$ of size $N = 2n\delta$ which is $(e, d\delta)$-depth-robust. In particular Theorem 4 shows that the CC of H is at least $\Omega(n^2/\log^3(n)) = \Omega(N^2/\log^7(N))$.

The above analysis uses the results of [AS15] as a blackbox. However, if we look into their construction of the graph $G_{n,\delta}$ can further improve on the analysis using our tools. In particular $G_{n,\delta}$ consists of a stack of $\log\log(n)$ depth-robust graphs taken from the construction of [MMV13]. Applying Theorem 4 to (say the first) of these layers we see that $\Pi^{\parallel}_{cc}(G_{n,\delta}) \geq \frac{cn^2}{(\log\log(n))^2}$ for some constant $c > 0$. Lemma 1 allows us to reduce the indegree $G_{n,\delta}$ at cost of $\delta \leq \log^2(n)\mathsf{polyloglog}(n)$. Thus, we obtain a constant indegree graph on N nodes with $\Pi^{\parallel}_{cc}(G) \geq \frac{cN^2}{\log^2(n)\mathsf{polyloglog}(n)} = \Omega\left(\frac{N^2}{\log^2 N}\right)$ when ignoring $\mathsf{polyloglog}$ terms.

Algorithm 1: $\mathsf{RGenPeb}\ (G, \overline{S}, \overline{d}, T)$

Arguments : $G = (V, E)$, $\overline{S} = (S_1, S_2, \ldots, S_k)$, $\overline{d} = d_1, \ldots, d_k$ and $T \subseteq V$ such that $S_1 \subseteq \ldots \subseteq S_k \subseteq V$ and for $n = |V|$ and $d_0 = \mathsf{depth}(G)$ it holds that $\forall i \in [k] : e_i d_{i-1} \geq n d_i$, $e_i = |S_i|$ and $d_i \geq \mathsf{depth}(G - S_i)$.

Local Variables: An (arbitrary) node-partition $D_1, \ldots, D_{2d_0} \subseteq V$ such that $|D_i| \leq \frac{n}{d_0}$ and $\mathsf{parents}\big(D_{i+1}\big) \subseteq \bigcup_{j\in[i]} D_j$

Output : $P_1, \ldots, P_{2d_0} \subseteq V$

$d \leftarrow \max\{j : T \bigcap D_j \neq \emptyset\}$ `// Need` $d \leq 2d_0$ `steps to pebble nodes in` T.
$P_0, \ldots, P_{2d_0-d} \leftarrow \emptyset$
if $k{=}0$ **then** `// Greedy Pebble`
    for $i \in [d_0]$ **do**
        $P_{2d_0-d+i} \leftarrow D_i \cup P_{2d_0-d+i-1}$
    end
    Return $P_1, \ldots, P_{2d_0}$
else
    $m \leftarrow e_2 d_1 / n$ `// Length of a light phase`
    for $t \in [2d_0]$ **do**
        $U_t \leftarrow \bigcup_{j\in[t]} D_j$ `// Everything pebbled by time` t.
    end
    for $c = [2d_0/m]$ **do**
        for $i \in [m]$ **do** `// Light Phase`
            $t \leftarrow (c-1)m + i$ `// Current time step.`
            $R_t \leftarrow \mathsf{parents}(U_{cm} \setminus U_t)$ `// Parents still needed.`
            $P_t \leftarrow D_t \cup (S_1 \cap P_{t-1}) \cup (R_k \cap P_{t-1})$
        end
        $X_{cm+m} \leftarrow \mathsf{parents}\left(U_{(c+1)m} \setminus U_{cm+1}\right) \cap U_{cm}$ `// For next light phase.`
        $G' \leftarrow G - (S_1 \cup (V \setminus U_{cm-2d_1+1}))$ `// Subgraph for balloon phase.`
        $X^-_{cm+m} \leftarrow (X_{cm+m} \setminus S_1) \cap U_{cm-2d_1}$ `// Target for recursive call.`
        $Q_1, \ldots, Q_{2d_1} \leftarrow \mathsf{RGenPeb}\left(G', (S_2, \ldots, S_k), (d_2, \ldots, d_k), X^-_{cm+m}\right)$
        for $i \in [m - 2d_1 + 1, m]$ **do** `// Balloon Phase`
            $t \leftarrow (c-1)m + i$ `// Current time step.`
            $P_t \leftarrow P_t \cup Q_i \cup (P_{t-1} \cap X_{c+1})$
        end
    end
end
Return $P_1, \ldots, P_{2d_0}$