arXiv:1705.05313v2 [cs.CR] 7 Jul 2017

Sustained Space Complexity

Joël Alwen* Jeremiah Blocki† Krzysztof Pietrzak*

July 11, 2017

Abstract

Memory-hard functions (MHF) are functions whose evaluation cost is dominated by memory cost. MHFs are egalitarian, in the sense that evaluating them on dedicated hardware (like FPGAs or ASICs) is not much cheaper than on off-the-shelf hardware (like x86 CPUs). MHFs have interesting cryptographic applications, most notably to password hashing and securing blockchains.

Alwen and Serbinenko [STOC'15] define the cumulative memory complexity (cmc) of a function as the sum (over all time-steps) of the amount of memory required to compute the function. They advocate that a good MHF must have high cmc. Unlike previous notions, cmc takes into account that dedicated hardware might exploit amortization and parallelism. Still, cmc has been critizised as insufficient, as it fails to capture possible time-memory trade-offs; as memory cost doesn't scale linearly, functions with the same cmc could still have very different actual hardware cost.

In this work we address this problem, and introduce the notion of sustained-memory complexity, which requires that any algorithm evaluating the function must use a large amount of memory for many steps. We construct functions (in the parallel random oracle model) whose sustained-memory complexity is almost optimal: our function can be evaluated using n steps and $O(n/\log(n))$ memory, in each step making one query to the (fixed-input length) random oracle, while any algorithm that can make arbitrary many parallel queries to the random oracle, still needs $\Omega(n/\log(n))$ memory for $\Omega(n)$ steps.

As has been done for various notions (including cmc) before, we reduce the task of constructing an MHFs with high sustained-memory complexity to proving pebbling lower bounds on DAGs. Our main technical contribution is the construction is a family of DAGs on n nodes with constant indegree with high "sustained-space complexity", meaning that any parallel black-pebbling strategy requires $\Omega(n/\log(n))$ pebbles for at least $\Omega(n)$ steps.

Along the way we construct a family of maximally "depth-robust" DAGs with maximum indegree $O(\log n)$, improving upon the construction of Mahmoody et al. [ITCS'13] which had maximum indegree $O\left(\log^2 n \cdot \mathsf{polylog}(\log n)\right)$.

1 Introduction

In cryptographic settings we typically consider tasks which can be done efficiently by honest parties, but are infeasible for potential adversaries. This requires an asymmetry in the capabilities of honest and dishonest parties. An example are trapdoor functions, where the honest party – who knows the secret trapdoor key – can efficiently invert the function, whereas a potential adversary – who does not have this key – cannot.

1.1 Moderately-Hard Functions

Moderately hard functions consider a setting where there's no asymmetry, or even worse, the adversary has more capabilities than the honest party. What we want is that the honest party can evaluate the function with some reasonable amount of resources, whereas the adversary should not be able to evaluate the function

*IST Austria
†Purdue University

at significantly lower cost. Moderately hard functions have several interesting cryptographic applications, including securing blockchain protocols and for password hashing.

An early proposal for password hashing is the "Password Based Key Derivation Function 2" (PBKDF2) [Kal00]. This function just iterates a cryptographic hash function like SHA1 several times (1024 is a typical value). Unfortunately PBKDF2 doesn't make for a good moderately hard function, as evaluating a cryptographic hash function on dedicated hardware like ASCIs (Application Specific Integrated Circuits) can be by several orders of magnitude cheaper in terms of hardware and energy cost than evaluating it on a standard x86 CPU. There have been several suggestions how to construct better, i.e., more "egalitarian", moderately hard functions. We discuss the most prominent suggestions below.

Memory-Bound Functions Abadi et al. [ABW03] observe that the time required to evaluate a function is dominated by the number of cache-misses, and these slow down the computation by about the same time over different architectures. They propose *memory-bound* functions, which are functions that will incur many expensive cache-misses (assuming the cache is not too big). They propose a construction which is not very practical as it requires a fairly large (larger than the cache size) incompressible string as input. Their function is then basically pointer jumping on this string. In subsequent work [DGN03] it was shown that this string can also be locally generated from a short seed.

Bandwidth-Hard Functions Recently Ren and Devadas [RD17] suggest the notion of *bandwidth-hard* functions, which is a refinement of memory-bound functions. A major difference being that in their model computation is not completely free, and this assumption – which of course is satisfied in practice – allows for much more practical solutions. They also don't argue about evaluation time as [ABW03], but rather the more important energy cost; the energy spend for evaluating a function consists of energy required for on chip computation and memory accesses, only the latter is similar on various platforms. In a bandwidth-hard function the memory accesses dominate the energy cost on a standard CPU, and thus the function cannot be evaluated at much lower energy cost on an ASICs as on a standard CPU.

Memory-Hard Function Whereas memory-bound and bandwidth-hard functions aim at being egalitarian in terms of time and energy, memory-hard functions (MHF), proposed by Percival [Per09], aim at being egalitarian in terms of hardware cost. A memory-hard function, in his definition, is one where the memory used by the algorithm, multiplied by the amount of time, is high, i.e., it has high space-time (ST) complexity. Moreover, parallelism should not help to evaluate this function at significantly lower cost by this measure. The rationale here is that the hardware cost for evaluating an MHF is dominated by the memory cost, and as memory cost does not vary much over different architectures, the hardware cost for evaluating MHFs is not much lower on ASICs than on standard CPUs.

Cumulative Memory Complexity Alwen and Serbinenko [AS15] observe that ST complexity misses a crucial point, amortization. A function might have high ST complexity because at some point during the evaluation the space requirement is high, but for most of the time a small memory is sufficient. As a consequence, ST complexity is not multiplicative: a function can have ST complexity C, but evaluating X instances of the function can be done with ST complexity much less than $X \cdot C$, so the amortized ST cost is much less than C. Alwen and Blocki [AB16, AB17] later showed that prominent MHF candidates such as Argon2i [BDK16], winner of the Password Hashing Competition [PHC] do not have high amortized ST complexity.

To address this issue, [AS15] put forward the notion of cumulative-memory complexity (cmc). The cmc of a function is the sum – over all time steps – of the memory required to compute the function by any algorithm. Unlike ST complexity, cmc is multiplicative.

Sustained-Memory Complexity Although cmc takes into account amortization and parallelism, it has been observed (e.g., [RD16, Cox16] that it still is not sufficient to guarantee egalitarian hardware cost. The reason is simple: if a function has cmc C, this could mean that the algorithm minimizing cmc uses some T time steps and C/T memory on average, but it could also mean it uses time $100 \cdot T$ and $C/100 \cdot T$ memory on average. In practice this can makes a huge difference because *memory cost doesn't scale linearly*. The length of the wiring required to access memory of size M grows like $\sqrt{M}$ (assuming a two dimensional layout of the circuit). This means for one thing, that – as we increase M – the latency of accessing the memory will grow as $\sqrt{M}$, and moreover the space for the wiring required to access the memory will grow like $M^{1.5}$.

The exact behaviour of the hardware cost as the memory grows is not crucial here, just the point that cmc misses to take into account that it's not linear. In this work we introduce the notion of sustained-memory complexity, which takes this into account. Ideally, we want a function which can be evaluated by a "naïve" sequential algorithm (the one used by the honest parties) in time T using a memory of size S where (1) S should be close to T and (2) any *parallel* algorithm evaluating the function must use memory S' for at least T' steps, where T' and S' should be not much smaller than T and S, respectively.

Property (1) is required so the memory cost dominates the evaluation cost already for small values of T. Property (2) means that even a parallel algorithm will not be able to evaluate the function at much lower cost; any parallel algorithm must make almost as many steps as the naïve algorithm during which the required memory is almost as large as the maximum memory S used by the naïve algorithm. So, the cost of the best parallel algorithm is similar to the cost of the naïve sequential one, even if we don't charge the parallel algorithm anything for all the steps where the memory is below S'.

Ren and Devadas [RD16] previously proposed the notion of "consistent memory hardness" which requires that any *sequential* evaluation algorithm must either use space S' for at least T' steps, or the algorithm must run for a long time e.g., $T \gg n^2$. Our notion of sustained-memory complexity strengthens this notion in that we consider *parallel* evaluation algorithms, and our guarantees are absolute e.g., even if a parallel attacker runs for a very long time he must still use memory S' for at least T' steps.

In this work we show that functions with almost optimal sustained-memory complexity exist in the random oracle model. We note that we must make some idealized assumption on our building block, like being a random oracle, as with the current state of complexity theory, we cannot even prove superlinear circuit lower-bounds for problems in $\mathcal{NP}$. For a given time T, our function uses maximal space $S = \Omega(T)$ for the naïve algorithm,[1] while any *parallel* algorithm must make at least $T' = \Omega(T)$ steps during which it uses $S' = \Omega(T/\log(T)) = \Omega(S/\log(S))$ of memory.

Graph Labelling The functions we construct are defined by DAGs. For a DAG $G_n = (V, E)$, we order the vertices $V = \{v_1, \ldots, v_n\}$ in some topological order (so if there's a path from i to j then $i < j$), with v_1 being the unique source, and v_n the unique sink of the graph. The function is now defined by G_n and the input specifies a random oracle H. The output is the label ℓ_n of the sink, where the label of a node v_i is recursively defined as $\ell_i = H(i, \ell_{p_1}, \ldots, \ell_{p_d})$ where $v_{p_1}, \ldots, v_{p_d}$ are the parents of v_i.

Pebbling Like many previous works, including [ABW03, RD17, AS15] discussed above, we reduce the task of proving lower bounds – in our case, on sustained memory complexity – for functions as just described, to proving lower bounds on some complexity of a pebbling game played on the underlying graph.

For example [RD17] define a cost function for the so called reb-blue pebbling game, which then implies lower bounds on the bandwidth hardness of the function defined over the corresponding DAG.

Most closely related to this work is [AS15], who show that a lower bound the so called sequential (or parallel) *cumulative (black) pebbling complexity* (cpc) of a DAG implies a lower bound on the sequential (or parallel) cumulative memory complexity (cmc) of the labelling function defined over this graph. Recently [ABP17] constructed a constant indegree family of DAGs with parallel cpc $\Omega(n^2/\log(n))$, which is optimal [AB16], and thus gives functions with optimal cmc.

The black pebbling game – as considered in cpc – goes back to [HP70, Coo73]. It is defined over a DAG $G = (V, E)$ and goes in round as follows. Initially all nodes are empty. In every round, the player can put a pebble on a node if all its parents contain pebbles (arbitrary many pebbles per round in the parallel game, just one in the sequential). Pebbles can be removed at any time. The game ends when a pebble is put on the sink. The cpc of such a game is the sum, over all time steps, of the pebbles placed on the graph.

[1]Recall that the naïve algorithm is sequential, so S must be in $O(T)$ as in time T the algorithm cannot even touch more than $O(T)$ memory.

The sequential (or parallel) cpc of G is the cpc of the sequential (or parallel) black pebbling strategy which minimizes this cost.

It's not hard to see that the sequential/parallel cpc of G directly implies the same upper bound on the sequential/parallel cmc of the graph labelling function, as to compute the function in the sequential/parallel random oracle model, one simply mimics the pebbling game, where putting a pebble on vertex v_i with parents $v_{p_1}, \ldots, v_{p_d}$ corresponds to the query $\ell_i \leftarrow H(i, \ell_{p_1}, \ldots, \ell_{p_d})$. And where one keeps a label ℓ_j in memory, as long as v_j is pebbled. If the labels $\ell_i \in \{0,1\}^w$ are w bits long, a cpc of p translates to cmc of $p \cdot w$.

More interestingly, the same has been shown to hold for interesting notions also for lower bounds. In particular, the *ex-post facto* argument [AS15] shows that any adversary who computes the label ℓ_n with high probability (over the choice of the random oracle H) with cmc of m, translates into a black pebbling strategy of the underlying graph with cpc almost m/w.

In this work we define the *sustained-space complexity* (ssc) of a sequential/parallel black pebbling game, and show that lower bounds on ssc translate to lower bounds on the sustained-memory complexity (smc) of the graph labelling function in the sequential/parallel random oracle model.

Consider a sequential (or parallel) black pebbling strategy (i.e., a valid sequence pebbling configurations where the last configuration contains the sink) for a DAG $G_n = (V, E)$ on $|V| = n$ vertices. For some space parameter $s \leq n$, the s-ssc of this strategy is the number of pebbling configurations of size at least s. The sequential (or parallel) s-ssc of G is the strategy minimizing this value. For example, if it's possible to pebble G using $s' < s$ pebbles (using arbitrary many steps), then its s-ssc is 0. Similarly as for csc vs cmc, an upper bound on s-ssc implies the same upper bound for $(w \cdot s)$-smc. In Section 5 we prove that also lower bounds on ssc translate to lower bounds on smc.

Thus, to construct a function with high parallel smc, it suffices to construct a family of DAGs with constant indegree and high parallel ssc. In Section 3 we construct such a family $\{G_n\}_{n \in \mathbb{N}}$ of DAGs where G_n has n vertices and has indegree 2, where $\Omega(n/\log(n))$-ssc is in $\Omega(n)$. This is basically the best we can hope for, as our bound on ssc trivially implies a $\Omega(n^2/\log(n))$ bound on csc, which is optimal for any constant indegree graph [AS15].

1.2 High Level Description of our Construction and Proof

Our construction of a family $\{G_n\}_{n \in \mathbb{N}}$ of DAGs with optimal ssc involves three building blocks:

The first building block is a construction of Paul et al. [PTC76] of a family of DAGs $\{G_n\}_{n \in \mathbb{N}}$ with $\mathsf{indeg}(G_n) = 2$ and space complexity $\Omega(n/\log n)$. More significantly for us they proved that for any sequential pebbling of G_n there is a time interval $[i, j]$ during which at least $\Omega(n/\log n)$ new pebbles are placed on sources of G_n and at least $\Omega(n/\log n)$ are always on the DAG. We extend the proof of Paul et al. [PTC76] to show that the same holds for any *parallel* pebbling of G_n. We can argue that $j - i = \Omega(n/\log n)$ for any sequential pebbling since it takes at least this many steps to place $\Omega(n/\log n)$ new pebbles on G_n. However, we stress that this argument does not apply to parallel pebblings so this does not directly imply anything about sustained space complexity for parallel pebblings.

To address this issue we introduce our second building block: a family of $\{G_n^\epsilon\}_{n \in \mathbb{N}}$ of extremely depth robust DAGs with $\mathsf{indeg}(G_n) = O(\log n)$ — for any constant $\epsilon > 0$ the DAG G_n^ϵ is (e, d)-depth robust for any $e + d \leq (1 - \epsilon)n$. We remark that our result improves upon the construction of Mahmoody et al.[MMV13] whose construction required $\mathsf{indeg}(G_n) = O\left(\log^2 n \mathsf{polylog}(\log n)\right)$ and may be of independent interest (e.g., our construction immediately yields a more efficient construction of proofs of sequential work [MMV13]). Our construction of G_n^ϵ is (essentially) the same as Erdos et al. [EGS75] albeit with much tighter analysis. By overlaying an extremely depth-robust DAG G_n^ϵ on top of the sources of G_n, the construction of Paul et al. [PTC76]. We can ensure that it takes $\Omega(n/\log n)$ steps to pebble $\Omega(n/\log n)$ sources of G_n. However, the resulting graph would have $\mathsf{indeg}(G_n) = O(\log n)$ and would have sustained space $\Omega(n/\log n)$ for at most $O(n/\log n)$ steps. By contrast, we want a n-node DAG G with $\mathsf{indeg}(G) = 2$ which requires space $\Omega(n/\log n)$ for at least $\Omega(n)$ steps.

Our final tool is to apply the indegree reduction lemma of Alwen et al. [ABP17] to $\{G_t^\epsilon\}_{t \in \mathbb{N}}$ to obtain a family of DAGs $\{D_t^\epsilon\}_{t \in \mathbb{N}}$ such that D_t has $\mathsf{indeg}\left(D_t^\epsilon\right) = 2$ and $2t\mathsf{indeg}(G_t) = O(t \log t)$ nodes. Each node in G_t is associated with a path of length $2\mathsf{indeg}(G_t)$ in D_t^ϵ and each path p in G_t corresponds to a path p' of length $|p'| \geq |p|\mathsf{indeg}(G_t)$ in D_t^ϵ. We can then overlay the DAG D_t^ϵ on top of the sources in G_n where $t = \Omega(n/\log n)$ is the number of sources in G_n. The final DAG has size $O(n)$ and we can then show that any legal parallel pebbling requires $\Omega(n)$ steps with at least $\Omega(n/\log n)$ pebbles on the DAG.

2 Preliminaries

In this section we introduce common notation, definitions and results from other work which we will be using. In particular the following borrows heavily from [ABP17, AT17].

2.1 Notation

We start with some common notation. Let $\mathbb{N} = \{0, 1, 2, \ldots\}$, $\mathbb{N}^+ = \{1, 2, \ldots\}$, and $\mathbb{N}_{\geq c} = \{c, c+1, c+2, \ldots\}$ for $c \in \mathbb{N}$. Further, we write $[c] := \{1, 2, \ldots, c\}$ and $[b, c] = \{b, b+1, \ldots, c\}$ where $c \geq b \in \mathbb{N}$. We denote the cardinality of a set B by $|B|$.

2.2 Graphs

The central object of interest in this work are directed acyclic graphs (DAGs). A DAG $G = (V, E)$ has *size* $n = |V|$. The indegree of node $v \in V$ is $\delta = \mathsf{indeg}(v)$ 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. If $\mathsf{indeg}(v) = 0$ then v is called a *source* node and if v has no outgoing edges it 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 $\mathsf{ancestors}_G(v) := \bigcup_{i \geq 1} \mathsf{parents}_G^i(v)$ to denote the set of all ancestors of v — here, $\mathsf{parents}_G^2(v) := \mathsf{parents}_G(\mathsf{parents}_G(v))$ denotes the grandparents of v and $\mathsf{parents}_G^{i+1}(v) := \mathsf{parents}_G\left(\mathsf{parents}_G^i(v)\right)$. 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 $\mathsf{ancestors}(\mathsf{sinks}(G)) = V$. The length of a directed path $p = (v_1, v_2, \ldots, v_z)$ in G 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 often consider the set of all DAGs of fixed size n $\mathbb{G}_n := \{G = (V, E) \ : \ |V| = n\}$ and the subset of those DAGs at most some fixed indegree $\mathbb{G}_{n,\delta} := \{G \in \mathbb{G}_n : \mathsf{indeg}(G) \leq \delta\}$. Finally, we denote the graph obtained from $G = (V, E)$ by removing nodes $S \subseteq V$ (and incident edges) by $G - S$ and we denote by $G[S] = G - (V \setminus S)$ the graph obtained by removing nodes $V \setminus S$ (and incident edges).

The following is an important combinatorial property of a DAG for this work.

Definition 2.1 (Depth-Robustness) *For $n \in \mathbb{N}$ and $e, d \in [n]$ a DAG $G = (V, E)$ is (e, d)-depth-robust if*

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

The following lemma due to Alwen et al. [ABP17] will be useful in our analysis. Since our statement of the result is slightly different from [ABP17] we include a proof in Appendix A for completeness.

Lemma 2.2 *[ABP17, Lemma 1] (***Indegree-Reduction***) Let $G = (V = [n], E)$ be a (e, d)-depth robust DAG on n nodes and let $\delta = \mathsf{indeg}(G)$. We can efficiently construct a DAG $G' = (V' = [2n\delta], E')$ on $2n\delta$ nodes with $\mathsf{indeg}(G') = 2$ such that for each path $p = (x_1, ..., x_k)$ in G there exists a corresponding path p' of length $\geq k\delta$ in G' $\left[\bigcup_{i=1}^{k} [2(x_i - 1)\delta + 1, 2x_i\delta]\right]$ such that $2x_i\delta \in p'$ for each $i \in [k]$. In particular, G' is $(e, d\delta)$-depth robust.*

2.3 Pebbling Models

The main computational models of interest in this work are the parallel (and sequential) pebbling games played over a directed acyclic graph. Below we define these models and associated notation and complexity measures. Much of the notation is taken from [AS15, ABP17].

Definition 2.3 (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. *A pebble can be added only if all its parents were pebbled 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 is 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 mostly be interested in the case where $T = \mathsf{sinks}(G)$ in which case we write $\mathcal{P}_G$ and $\mathcal{P}^{\parallel}_G$.

Definition 2.4 (Pebbling Complexity) *The standard notions of* time*,* space*,* space-time *and* cumulative (pebbling) complexity *(cc) of a pebbling $P = \{P_0, \ldots, P_t\} \in \mathcal{P}^{\parallel}_G$ are defined to be:*

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

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

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

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

The following defines a sequential pebbling obtained naturally from a parallel one by adding each new pebble on at a time.

Definition 2.5 *Given a DAG G and $P = (P_0, \ldots, P_t) \in \mathcal{P}^{\parallel}_G$ the* sequential transform $\mathsf{seq}(P) = P' \in \Pi_G$ *is defined as follows: Let difference $D_j = P_i \setminus P_{i-1}$ and let $a_i = |P_i \setminus P_{i-1}|$ be the number of new pebbles placed on G_n at time i. Finally, let $A_j = \sum_{i=1}^{j} a_i$ ($A_0 = 0$) and let $D_j[k]$ denote the k^{th} element of D_j (according to some fixed ordering of the nodes). We can construct $P' = \left(P'_1, \ldots, P'_{A_t}\right) \in \mathcal{P}(G_n)$ as follows: (1) $P'_{A_i} = P_i$ for all $i \in [0, t]$, and (2) for $k \in [1, a_{i+1}]$ let $P'_{A_i+k} = P'_{A_i+k-1} \cup D_j[k]$.*

If easily follows from the definition that the parallel and sequential space complexities differ by at most a multiplicative factor of 2.

Lemma 2.6 *For any DAG G and $P \in \mathcal{P}^{\parallel}_G$ it holds that $\mathsf{seq}(P) \in \mathcal{P}_G$ and $\Pi_s(\mathsf{seq}(P)) \leq 2 * \Pi^{\parallel}_s(P)$. In particular $\Pi_s(G)/2 \leq \Pi^{\parallel}_s(G)$.*

Proof. Let $P \in \mathcal{P}^{\parallel}_G$ and $P' = \mathsf{seq}(P)$. Suppose P' is not a legal pebbling because $v \in V$ was illegally pebbled in P'_{A_i+k}. If $k = 0$ then $\mathsf{parents}_G(v) \not\subseteq P'_{A_{i-1}+a_i-1}$ which implies that $\mathsf{parents}_G(v) \not\subseteq P_{i-1}$ since $P_{i-1} \subseteq P'_{A_{i-1}+a_i-1}$. Moreover $v \in P_i$ so this would mean that also P illegally pebbles v at time i. If instead, $k > 1$ then $v \in P_{i+1}$ but since $\mathsf{parents}_G(v) \not\subseteq P'_{A_i+k-1}$ it must be that $\mathsf{parents}_G(v) \not\subseteq P_i$ so P must have pebbled v illegally at time $i+1$. Either way we reach a contradiction so P' must be a legal pebbling of G. To see that P' is complete note that $P_0 = P'_{A_0}$. Moreover for any sink $u \in V$ of G there exists time $i \in [0,t]$ with $u \in P_i$ and so $u \in P'_{A_i}$. Together this implies $P' \in \mathcal{P}_G$.

Finally, it follows by inspection that for all $i \geq 0$ we have $|P'_{A_i}| = |P_i|$ and for all $0 < k < a_i$ we have $|P'_{A_i+k}| \leq |P_i| + |P_{i+1}|$ which implies that $\Pi_s(P') \leq 2 * \Pi_s^{\parallel}(P)$. □

New to this work is the following notion of sustained-space complexity.

Definition 2.7 (Sustained Space Complexity) *For $s \in \mathbb{N}$ the* s-sustained-space *(s-ss) complexity of a pebbling $P = \{P_0, \ldots, P_t\} \in \mathcal{P}_G^{\parallel}$ is:*

$$\Pi_{ss}(P,s) = |\{i \in [t] : |P_i| \geq s\}|.$$

More generally, the sequential and parallel s-sustained space complexities of G are defined as

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

As before, when $T = \mathsf{sinks}(G)$ we simplify notation and write $\Pi_{ss}(G,s)$ and $\Pi_{ss}^{\parallel}(G,s)$.

Remark 1 *(On Amortization) An astute reader may observe that $\Pi_{ss}^{\parallel}$ is not amortizable. In particular, if we let $G^{\bigotimes m}$ denotes the graph which consists of m independent copies of G then we may have $\Pi_{ss}^{\parallel}\left(G^{\bigotimes m}, s\right) \ll m\Pi_{ss}^{\parallel}(G,s)$. However, we observe that the issue can be easily corrected by defining the* amortized s-sustained-space *complexity of a pebbling $P = \{P_0, \ldots, P_t\} \in \mathcal{P}_G^{\parallel}$:*

$$\Pi_{am,ss}(P,s) \sum_{i=1}^{t} \left\lfloor \frac{|P_i|}{s} \right\rfloor .$$

In this case we have $\Pi_{am,ss}^{\parallel}\left(G^{\bigotimes m}, s\right) = m\Pi_{am,ss}^{\parallel}(G,s)$ where $\Pi_{am,ss}^{\parallel}(G,s) \doteq \min_{P \in \mathcal{P}_{G,\mathsf{sinks}(G)}^{\parallel}} \Pi_{am,ss}(P,s)$. We also remark that s-sustained-space *complexity is a strictly stronger guarantee than* amortized s-sustained-space *since $\Pi_{ss}^{\parallel}(G,s) \leq \Pi_{am,ss}^{\parallel}(G,s)$. Thus, all of our lower bounds from $\Pi_{ss}^{\parallel}$ also hold with respect to $\Pi_{am,ss}^{\parallel}$.*

The following shows that the indegree of any graph can be reduced down to 2 with out loosing too much in the parallel sustained space complexity. The technique is similar the indegree reduction for cumulative complexity in [AS15]. The proof is in Appendix A.

Lemma 2.8 (Indegree Reduction for Parallel Sustained Space)

$$\forall G \in \mathbb{G}_{n,\delta}, \quad \exists H \in \mathbb{G}_{n',2} \textit{ such that } \forall s \geq 0 \ \ \Pi_{ss}^{\parallel}(H, s/(\delta-1)) = \Pi_{ss}^{\parallel}(G,s) \textit{ where } n' \in [n, \delta n].$$

3 A Graph with Optimal Sustained Space Complexity

In this section we construct and analyse a graph with very high sustained space complexity by modifying the graph of [PTC76] using the graph of [EGS75]. Theorem 3.1, our main theorem, states that there is a family of constant indegree DAGs $\{G_n\}_{n=1}^{\infty}$ with maximum possible sustained space $\Pi_{ss}\left(G_n, \Omega(n/\log n)\right) = \Omega(n)$.

Theorem 3.1 *For some constants $c_4, c_5 > 0$ there is a family of DAGs $\{G_n\}_{n=1}^{\infty}$ with $\mathsf{indeg}\left(G_n\right) = 2$, $O(n)$ nodes and $\Pi_{ss}^{\parallel}\left(G_n, c_4 n/\log n\right) \geq c_5 n$.*

Remark 2 *We observe that Theorem 3.1 is essentially optimal in an asymptotic sense. Hopcroft et al. [HPV77] showed that any DAG G_n with $\mathsf{indeg}(G_n) = O(1)$ can be pebbled with space at most $\Pi_s^{\parallel}(G_n) = O\left(n/\log n\right)$. Thus, $\Pi_{ss}\left(G_n, s_n = \omega\left(n/\log n\right)\right) = 0$ for any DAG G_n with $\mathsf{indeg}(G_n) = O(1)$ since $s_n > \Pi_s(G_n)$.* [2]

We now overview the key technical ingredients in the proof of Theorem 3.1.

[2]Furthermore, even if we restrict our attention to pebblings which finish in time $O(n)$ we still have $\Pi_{ss}\left(G_n, f(n)\right) \leq g(n)$

Technical Ingredient 1: High Space Complexity DAGs

The first key building blocks is a construction of Paul et al. [PTC76] of a family of n node DAGs $\{G_n\}_{n=1}^{\infty}$ with space complexity $\Pi_s(G_n) = \Omega(n/\log n)$ and $\mathsf{indeg}(G_n) = 2$. Lemma 2.6 implies that $\Pi_s^{\parallel}(G_n) = \Omega(n/\log n)$ since $\Pi_s(G_n)/2 \leq \Pi_s^{\parallel}(G_n)$. However, we stress that this does not imply that the sustained space complexity of G_n is large. In fact, by inspection one can easily verify that $\mathsf{depth}(G_n) = O(n/\log n)$ so we have $\Pi_{ss}(G_n, s) \leq O(n/\log n)$ for any space parameter $s > 0$. Nevertheless, one of the core lemmas from [PTC76] will be very useful in our proofs. In particular, G_n contains $O(n/\log n)$ source nodes and [PTC76] proved that for any sequential pebbling $P = (P_0, \ldots, P_t) \in \Pi_G$ we can find an interval $[i,j] \subseteq [t]$ during which $\Omega(n/\log n)$ sources are (re)pebbled and at least $\Omega(n/\log n)$ pebbles are always on the graph — see Theorem A.3 in Appendix A for a formal statement of their original result.

As we show in Theorem 3.2 the same claim holds for all parallel pebblings $P \in \Pi_{G_n}^{\parallel}$. Since Paul et al. [PTC76] only considered sequential black pebblings we include the straightforward proof of Theorem 3.2 in Appendix A for completeness. Briefly, to prove Theorem 3.2 we simply consider the sequential transform $\mathsf{seq}(P) = (Q_0, \ldots, Q_{t'}) \in \Pi_{G_n}$ of the parallel pebbling P. Since $\mathsf{seq}(P)$ is sequential we can find an interval $[i', j'] \subseteq [t']$ during which $\Omega(n/\log n)$ sources are (re)pebbled and at least $\Omega(n/\log n)$ pebbles are always on the graph G_n. We can then translate $[i', j']$ to a corresponding interval $[i,j] \subseteq [t]$ during which the same properties hold for P.

Theorem 3.2 *There is a family of DAGs $\{G_n = (V_n = [n], E_n)\}_{n=1}^{\infty}$ with $\mathsf{indeg}\,(G_n) = 2$ with the property that for some positive constants $c_1, c_2, c_3 > 0$ such that for each $n \geq 1$ the set $S = \{v \in [n] \ : \ \mathsf{parents}(v) = \emptyset\}$ of sources of G_n has size $|S| \leq c_1 n/\log n$ and for any legal pebbling $P = (P_1, \ldots, P_t) \in \mathcal{P}_{G_n}^{\parallel}$ there is an interval $[i,j] \subseteq [t]$ such that (1) $\left| S \cap \bigcup_{k=i}^{j} P_k \setminus P_{i-1} \right| \geq c_2 n/\log n$ i.e., at least $c_2 n/\log n$ nodes in S are (re)pebbled during this interval, and (2) $\forall k \in [i,j], |P_k| \geq c_3 n/\log n$ i.e., at least $c_3 n/\log n$ pebbles are always on the graph.*

One of the key remaining challenges to establishing high sustained space complexity is that the interval $[i,j]$ we obtain from Theorem 3.2 might be very short for parallel black pebblings. For sequential pebblings it would take $\Omega(n/\log n)$ steps to (re)pebble $\Omega(n/\log n)$ source nodes since we can add at most one new pebble in each round. However, for parallel pebblings we cannot rule out the possibility that all $\Omega(n/\log n)$ sources were pebbled in a single step!

A first attempt at a fix is to modify G_n by overlaying a path of length $\Omega(n)$ on top of these $\Omega(n/\log n)$ source nodes to ensure that the length of the interval $j - i + 1$ is sufficiently large. The hope is that it will take now at least $\Omega(n)$ steps to (rep)pebble any subset of $\Omega(n/\log n)$ of the original sources since these nodes will be connected by a path of length $\Omega(n)$. However, we do not know what the pebbling configuration looks like at time $i-1$. In particular, if P_{i-1} contained just $\sqrt{n}$ of the nodes on this path then the it would be possible to (re)pebble all nodes on the path in at most $O\left(\sqrt{n}\right)$ steps. This motivates our second technical ingredient: extremely depth-robust graphs.

Technical Ingredient 2: Extremely Depth-Robust Graphs

Our second ingredient is a family $\{D_n^{\epsilon}\}_{n=1}^{\infty}$ of highly depth-robust DAGs with n nodes and $\mathsf{indeg}(D_n) = O(\log n)$. In particular, D_n^{ϵ} is (e,d)-depth robust for *any* $e + d \leq n(1-\epsilon)$. We show how to construct such a family $\{D_n^{\epsilon}\}_{n=1}^{\infty}$ for for any constant $\epsilon > 0$ in Section 4. Assuming for now that such a family exists we can overlay D_m over the $m \leq c_1 n/\log n$ sources of G_n. Since D_m^{ϵ} is *highly* depth-robust it will take at least $c_2 n/\log n - \epsilon m \geq c_2 n/\log n - \epsilon c_1 n/\log n = \Omega(n/\log n)$ steps to pebble these $c_2 n/\log n$ sources during the interval $[i,j]$.

whenever $f(n)g(n) = \omega\left(\frac{n^2 \log\log n}{\log n}\right)$ and $\mathsf{indeg}(G_n) = O(1)$. In particular, Alwen and Blocki [AB16] showed that for any G_n with $\mathsf{indeg}(G_n) = O(1)$ then there is a pebbling $P = (P_0, \ldots, P_n) \in \Pi_{G_n}^{\parallel}$ with $\Pi_{cc}^{\parallel}(P) \leq O\left(\frac{n^2 \log\log n}{\log n}\right)$. By contrast, the generic pebbling [HPV77] of any DAG with $\mathsf{indeg} = O(1)$ in space $O\left(n/\log n\right)$ can take exponentially long.

Overlaying D_m^ϵ over the $m = O(n/\log(n))$ sources of G_n yields a DAG G with $O(n)$ nodes, $\mathsf{indeg}(G) = O(\log n)$ and $\Pi^{\parallel}_{ss}(G, c_4 n/\log n) \geq c_5 n/\log n$ for some constants $c_4, c_5 > 0$. While this is progress it is still a weaker result than Theorem 3.1 which promised a DAG G with $O(n)$ nodes, $\mathsf{indeg}(G) = 2$ and $\Pi^{\parallel}_{ss}(G, c_4 n/\log n) \geq c_5 n$ for some constants $c_4, c_5 > 0$. Thus, we need to introduce a third technical ingredient: indegree reduction.

Technical Ingredient 3: Indegree Reduction

To ensure $\mathsf{indeg}(G) = 2$ we instead apply indegree reduction algorithm from Lemma 2.2 to D_m^ϵ to obtain a graph J_m^ϵ with $2m\delta = O(n)$ nodes $[2\delta m]$ and $\mathsf{indeg}(J_m^\epsilon) = 2$ before overlaying — here $\delta = \mathsf{indeg}(D_m^\epsilon)$. We then associate the m sources of G_n with the nodes $\{2\delta v \ : v \in [m]\}$ in J_m^ϵ. J_m^ϵ is $(e, \delta d)$-depth robust for any $e + d \leq (1-\epsilon)m$, which seems to suggests that it will take $\Omega(n)$ steps to (re)pebble $c_2 n/\log n$ sources during the interval. However, we still run into the same problem: In particular, suppose that at some point in time k we can find a set $T \subseteq \{2v\delta : v \in [m]\} \setminus P_k$ with $|T| \geq c_2 n/\log n$ (e.g., a set of sources in G_n) such that the longest path running through T in $J_m^\epsilon - P_k$ has length at most $c_5 n$. If the interval $[i, j]$ starts at time $i = k+1$ then cannot ensure that it will take time $\geq c_5 n$ to (re)pebble these $c_2 n/\log n$ source nodes.

Claim 3.3 addresses this challenge directly. If such a problematic time k exists then Claim 3.3 implies that we must have $\Pi^{\parallel}_{ss}(P, \Omega(n/\log n))) \geq \Omega(n)$. At a high level the argument proceeds as follows: suppose that we find such a problem time k along with a set $T \subseteq \{2v\delta : v \in [m]\} \setminus P_k$ with $|T| \geq c_2 n/\log n$ such that $\mathsf{depth}(J_m^\epsilon[T]) \leq c_5 n$. Then for any time $r \in [k - c_5 n, k]$ we know that the the length of the longest path running through T in $J_m^\epsilon - P_r$ is at most $\mathsf{depth}(J_m^\epsilon[T] - P_r) \leq c_5 n + (k - r) \leq 2c_5 n$ since the depth can decrease by at most one each round. We can then use the extreme depth-robustness of D_m^ϵ and the construction of J_m^ϵ to argue that $|P_r| = \Omega(n/\log n)$ for each $r \in [k - c_5 n, k]$. Finally, if no such problem time k exists then the interval $[i, j]$ we obtain from Theorem 3.2 must have length at least $i - j \geq c_5 n$. In either case we have $\Pi^{\parallel}_{ss}(P, \Omega(n/\log n))) \geq \Omega(n)$.

Proof of Theorem 3.1. We begin with the family of DAGs $\{G_n\}_{n=1}^{\infty}$ from Theorem 3.2 and Theorem A.3. Fixing $G_n = ([n], E_n)$ we let $S = \{v \in [n] : \mathsf{parents}(v) = \emptyset\} \subseteq V$ denote the sources of this graph and we let $c_1, c_2, c_3 > 0$ be the constants from Theorem 3.2. Let $\epsilon \leq c_2/(4c_1)$. By Theorem 4.1 we can find a depth-robust DAG $D_{|S|}^\epsilon$ on $|S|$ nodes which is $(a|S|, b|S|)$-DR for any $a + b \leq 1 - \epsilon$ with indegree $c' \log n \leq \delta = \mathsf{indeg}(D) \leq c'' \log(n)$ for some constants c', c''. We let $J_{|S|}^\epsilon$ denote the indegree reduced version of $D_{|S|}^\epsilon$ from Lemma 2.2 with $2|S|\delta = O(n)$ nodes and $\mathsf{indeg} = 2$. To obtain our DAG G from J_n^ϵ and G_n we associate each of the S nodes $2v\delta$ in J_n^ϵ with one of the nodes in S. We observe that G has at most $2|S|\delta + n = O(n)$ nodes and that $\mathsf{indeg}(G) \leq \max\{\mathsf{indeg}(G_n), \mathsf{indeg}(J_n^\epsilon)\} = 2$ since we do not increase the indegree of any node in J_n^ϵ when overlaying and in G_n do not increase the indegree of any nodes other that sources S (which may now have indegree 2 in J_n^ϵ).

Let $P = (P_0, \ldots, P_t) \in \mathcal{P}^{\parallel}_G$ be given and observe that by restricting $P_i' = P_i \cap V(G_n) \subseteq P_i$ we have a legal pebbling $P' = (P_0', \ldots, P_t') \in \mathcal{P}^{\parallel}_{G_n}$ for G_n. Thus, by Theorem 3.2 we can find an interval $[i, j]$ during which at least $c_2 n/\log n$ nodes in S are (re)pebbled and $\forall k \in [i, j]$ we have $|P_k| \geq c_3 n/\log n$. We use $T = S \cap \bigcup_{x=i}^{j} P_x - P_{i-1}$ to denote the source nodes of G_n that are (re)pebbled during the interval $[i, j]$. We now set $c_4 = c_2/4$ and $c_5 = c_2 c'/4$ and consider two cases:

Case 1: We have $\mathsf{depth}(\mathsf{ancestors}_{G-P_i}(T)) \geq |T|\delta/4$. In other words at time i there is an unpebbled path of length $\geq |T|\delta/4$ to some node in T. In this case, it will take at least $j - i \geq |T|\delta/4$ steps to pebble T so we have $|T|\delta/4 = \Omega(n)$ steps with at least $c_3 n/\log n$ pebbles. Because $c_5 = c_2 c'/4$ it follows that $|T|\delta/4 \geq c_2 c' n \geq c_5 n$. Finally, since $c_4 \leq c_2$ we have $\Pi^{\parallel}_{ss}(G_n, c_4 n/\log n) \geq c_5 n$.

Case 2: We have $\mathsf{depth}(\mathsf{ancestors}_{G-P_i}(T)) < |T|\delta/4$. In other words at time i there is no unpebbled path of length $\geq |T|\delta/4$ to any node in T. Now Claim 3.3 directly implies that $\Pi^{\parallel}_{ss}(P, |T| - \epsilon|S| - |T|/2)) \geq \delta|T|/4$. This in turn implies that $\Pi^{\parallel}_{ss}(P, (c_2/2)n/(\log n) - \epsilon|S|) \geq \delta c_2 n/(2\log n)$. We observe that $\delta c_2 n/(2\log n) \geq c_5 n$ since, we have $c_5 = c_2 c'/4$. We also observe that $(c_2/2)n/\log n - \epsilon|S| \geq (c_2/2 - \epsilon c_1)n/\log n \geq (c_2/2 - c_2/4)n/\log n \geq c_2 n/(4\log n) = c_4 n$ since $|S| \leq c_1 n/\log n$, $\epsilon \leq c_2/(4c_1)$ and $c_4 = c_2/4$. Thus, in this case we also have $\Pi^{\parallel}_{ss}(P, c_4 n/\log n) \geq c_5 n$. $\Box$

Claim 3.3 *Let D^{ϵ}_n be an DAG with nodes $V(D^{\epsilon}_n) = [n]$, indegree $\delta = \mathsf{indeg}(D^{\epsilon}_n)$ that is (e,d)-depth robust for all $e, d > 0$ such that $e + d \leq (1-\epsilon)n$, let J^{ϵ}_n be the indegree reduced version of D^{ϵ}_n from Lemma 2.2 with 2δ nodes and $\mathsf{indeg}(J^{\epsilon}_n) = 2$, let $T \subseteq [n]$ and let $P = (P_1, \ldots, P_t) \in \mathcal{P}^{\parallel}_{J^{\epsilon}_n, \emptyset}$ be a (possibly incomplete) pebbling of J^{ϵ}_n. Suppose that during some round i we have $\mathsf{depth}\left(\mathsf{ancestors}_{J^{\epsilon}_n - P_i}\left(\bigcup_{v \in T}\{2\delta v\}\right)\right) \leq c\delta|T|$ for some constant $0 < c < \frac{1}{2}$. Then $\Pi^{\parallel}_{ss}(P, |T| - \epsilon n - 2c|T|)) \geq c\delta|T|$.*

Proof. For each time step r we let $H_r = \mathsf{ancestors}_{J^{\epsilon}_n - P_r}\left(\bigcup_{v\in T}\{2\delta v\}\right)$ and let $k < i$ be the last pebbling step before i during which $\mathsf{depth}(G_k) \geq 2c|T|\delta$. Observe that $k - i \geq \mathsf{depth}(H_k) - \mathsf{depth}(H_i) \geq cn\delta$ since we can decrease the length of any unpebbled path by at most one in each pebbling round. We also observe that $\mathsf{depth}(H_k) = c|T|\delta$ since $\mathsf{depth}(H_k) - 1 \leq \mathsf{depth}(H_{k+1}) < 2c|T|\delta$.

Let $r \in [k, i]$ be given then, by definition of k, we have $\mathsf{depth}(H_r) \leq 2c|T|\delta$. Let $P'_r = \{v \in V(D^{\epsilon}_n) : P_r \cap [2\delta(v-1)+1, 2\delta v] \neq \emptyset\}$ be the set of nodes $v \in [n] = V(D^{\epsilon}_n)$ such that the corresponding path $2\delta(v-1)+1, \ldots, 2\delta v$ in J^{ϵ}_n contains at least one pebble at time r. By depth-robustness of D^{ϵ}_n we have

$$\mathsf{depth}(D^{\epsilon}_n[T] - P'_r) \geq |T| - |P'_r| - \epsilon n \ . \tag{1}$$

On the other hand, exploiting the properties of the indegree reduction from Lemma 2.2, we have

$$\mathsf{depth}(D^{\epsilon}_n[T] - P'_r)\,\delta \leq \mathsf{depth}(H_r) \leq 2c|T|\delta \ . \tag{2}$$

Combining Equation 1 and Equation 2 we have

$$|T| - |P'_r| - \epsilon n \leq \mathsf{depth}(D^{\epsilon}_n[T] - P'_r) \leq 2c|T| \ .$$

It immediately follows that $|P_r| \geq |P'_r| \geq |T| - 2c|T| - \epsilon n$ for each $r \in [k, i]$ and, therefore, $\Pi^{\parallel}_{ss}(P, |T| - \epsilon n - 2c|T|) \geq c\delta|T|$. $\Box$

Remark 3 *(On the Explicitness of Our Construction) Our construction of a family of DAGs with high sustained space complexity is explicit in the sense that there is a probabilistic polynomial time algorithm which, except with very small probability, outputs an n node DAG G that has high sustained space complexity. In particular, Theorem 3.1 relies on an explicit construction of [PTC76], and the extreme depth-robust DAGs from Theorem 4.1. The construction of [PTC76] in turn uses an object called superconcentrators. Since we have explicit constructions of superconcentrators [**?**] the construction of [PTC76] can be made explicit. While the proof of the existence of a family of extremely depth-robust DAGs is not explicit the proof uses a probabilistic argument and can be adapted to obtain a probabilistic polynomial time which, except with very small probability, outputs an n node DAG G that is extremely depth-robust. In practice, however it is also desirable to ensure that there is a local algorithm which, on input v, computes the set $\mathsf{parents}(v)$ in time $\mathsf{polylog}(n)$. It is an open question whether any DAG G with high sustained space complexity allows for highly efficient computation of the set $\mathsf{parents}(v)$.*

4 Better Depth-Robustness

In this section we improve on the original analysis of Erdos et al. [EGS75], who constructed a family of DAGs with $\mathsf{indeg} = O(\log n)$ that is $(e = \Omega(n), d = \Omega(n))$-depth robust. Such a DAG G_n is not sufficient for us since we require that the subgraph $G_n[T]$ is also highly depth robust for any sufficiently large subset $T \subseteq V_n$ of nodes e.g., for any T such that $|T| \geq n/1000$. For any fixed constant $\epsilon > 0$ [MMV13] constructs a family of DAGs $\{G^{\epsilon}_n\}_{n=1}^{\infty}$ which is $(\alpha n, \beta n)$-depth robust for any positive constants α, β such that $\alpha + \beta \leq 1 - \epsilon$ but their construction has indegree $O\left(\log^2 n \cdot \mathsf{polylog}(\log n)\right)$. By contrast our results in the previous section assumed the the existence of such a family of DAGs with $\mathsf{indeg}(G_n) = O(\log n)$.

In fact our family of DAGs is essentially the same as [EGS75] with one minor modification to make the construction for for all $n > 0$. Our contribution in this section is an improved analysis which shows that the family of DAGs $\{G_n^\epsilon\}_{n=1}^\infty$ with indegree $O(\log n)$ is $(\alpha n, \beta n)$-depth robust for any positive constants α, β such that $\alpha + \beta \leq 1 - \epsilon$.

We remark that if we allow our family of DAGs to have $\mathsf{indeg}(G_n) = O(\log n \log^* n)$ then we can eliminate the dependence on ϵ entirely. In particular, we can construct a family of DAGs $\{G_n^\epsilon\}_{n=1}^\infty$ with $\mathsf{indeg}(G_n) = O(\log n \log^* n)$ such that for any positive constants such that $\alpha + \beta < 1$ the DAG G_n is $(\alpha n, \beta n)$-depth robust for all suitably large n.

Theorem 4.1 *Fix $\epsilon > 0$ then there exists a family of DAGs $\{G_n^\epsilon\}_{n=1}^\infty$ with $\mathsf{indeg}(G_n^\epsilon) = O(\log n)$ that is $(\alpha n, \beta n)$-depth robust for any constants α, β such that $\alpha + \beta < 1 - \epsilon$.*

The proof of Theorem 4.1 relies on Lemma 4.2, Lemma 4.3 and Lemma 4.4. We say that G is a δ-local expander if for every node $x \in [n]$ and every $r \leq x, n - x$ and every pair $A \subseteq I_r(x) \doteq \{x - r - 1, \ldots, x\}, B \subseteq I_r^*(x) \doteq \{x + 1, \ldots, x + r\}$ with size $|A|, |B| \geq \delta r$ we have $A \times B \cap E \neq \emptyset$ i.e., there is a directed edge from some node in A to some node in B. Lemma 4.2 says that for any constant $\delta > 0$ we can construct a family of DAGs $\{G_n^\delta\}_{n=1}^\infty$ with $\mathsf{indeg} = O(\log n)$ such that each G_n^δ is a δ-local expander. Lemma 4.2 essentially restates [EGS75, Claim 1] except that we require that G_n is a δ-local expander for *all* $n > 0$ instead of for n sufficiently large. Since we require a (very) minor modification to achieve δ-local expansion for *all* $n > 0$ we sketch the proof of Lemma 4.2 in Appendix A for completeness.

Lemma 4.2 *[EGS75] Let $\delta > 0$ be a fixed constant then there is a family of DAGs $\{G_n^\delta\}_{n=1}^\infty$ with $\mathsf{indeg} = O(\log n)$ such that each G_n^δ is a δ-local expander.*

While Lemma 4.2 essentially restates [EGS75, Claim 1], Lemma 4.3 and Lemma 4.4 improve upon the analysis of [EGS75]. We say that a node $x \in [n]$ is γ-good under a subset $S \subseteq [n]$ if for all $r > 0$ we have $|I_r(x) \backslash S| \geq \gamma |I_r(x)|$ and $|I_r^*(x) \backslash S| \geq \gamma |I_r^*(x)|$. Lemma 4.3 is similar to [EGS75, Claim 3], which also states that all γ-good nodes are connected by a directed path in $G - S$. However, we stress that the argument of [EGS75, Claim 3] requires that $\gamma \geq 0.5$ while Lemma 4.3 has no such restriction. This is crucial to prove Theorem 4.1 where we will select γ to be very small.

Lemma 4.3 *Let $G = (V = [n], E)$ be a δ-local expander and let $x < y \in [n]$ both be γ-good under $S \subseteq [n]$ then if $\delta < \min\{\gamma/2, 1/4\}$ then there is a directed path from node x to node y in $G - S$.*

Lemma 4.4 shows that *almost all* of the nodes in $G - S$ are γ-good. It immediately follows that $G_n - S$ contains a directed path running through *almost all* of the nodes $[n] \setminus S$. While Lemma 4.4 may appear similar to [EGS75, Claim 2] at first glance, we again stress one crucial difference. The proof of [EGS75, Claim 2] is only sufficient to show that at least $n - 2|S|/(1-\gamma) \geq n - 2|S|$ nodes are γ-good. At best this would allow us to conclude that G_n is $(e, n - 2e)$-depth robust. Together Lemma 4.4 and Lemma 4.3 imply that if G_n is a δ-local expander ($\delta < \min\{\gamma/2, 1/4\}$) then G_n is $\left(e, n - e\frac{1+\gamma}{1-\gamma}\right)$-depth robust.

Lemma 4.4 *For any DAG $G = ([n], E)$ and any subset $S \subseteq [n]$ of nodes at least $n - |S|\frac{1+\gamma}{1-\gamma}$ of the remaining nodes in G are γ-good with respect to S.*

Proof of Theorem 4.1. By Lemma 4.2, for any $\delta > 0$, there is a family of DAGs $\{J_n^\delta\}_{n=1}^\infty$ with $\mathsf{indeg} = O(\log n)$ such that for each $n \geq 1$ the DAG J_n^δ is a δ-local expander. Given $\epsilon \in (0, 1]$ we will set $G_n^\epsilon = J_n^\delta$ with $\delta = \epsilon/10 < 1/4$ so that G_n^ϵ is a $(\epsilon/10)$-local expander. We also set $\gamma = \epsilon/4 > 2\delta$. Let $S \subseteq V_n$ of size $|S| \leq e$ be given. Then by Lemma 4.4 at least $n - e\frac{1+\gamma}{1-\gamma}$ of the nodes are γ-good and byLemma 4.3 there is a path connecting all γ-good nodes in $G - S$. Thus, the DAG G_n^ϵ is $\left(e, n - e\frac{1+\gamma}{1-\gamma}\right)$-depth robust for any $e \leq n$. In particular, if $\alpha = e/n$ and $\beta = 1 - \alpha\frac{1+\gamma}{1-\gamma}$ then the graph is $(\alpha n, \beta n)$-depth robust. Finally we verify that

$$n - \alpha n - \beta n = -e + e\alpha\frac{1+\gamma}{1-\gamma} = e\frac{2\gamma}{1-\gamma} \leq n\frac{\epsilon}{2 - \epsilon/2} \leq \epsilon n \ .$$

$\square$

The proof of Lemma 4.3 follows by induction on the distance $|y-x|$ between γ-good nodes x and y. Our proof extends a similar argument from [EGS75] with one important difference. [EGS75] argued inductively that for each good node x and for each $r>0$ over half of the nodes in $I_r^*(x)$ are reachable from x and that x can be reached from over half of the nodes in $I_r(x)$ — this implies that y is reachable from x since there is at least one node $z \in I_{|y-x|}^*(x) = I_{|y-x|}(y)$ such that z can be reached from x and y can be reached from z in $G-S$. Unfortunately, this argument inherently requires that $\gamma \geq 0.5$ since otherwise we may have at least $|I_r^*(x) \cap S| \geq (1-\gamma)r$ nodes in the interval $I_r(x)$ that are not reachable from x. To get around this limitation we instead show, see Claim 4.5, that more than half of the nodes in the set $I_r^*(x) \setminus S$ are reachable from x and that more than half of the nodes in the set $I_r(x) \setminus S$ are reachable from x — this still suffices to show that x and y are connected since by the pigeonhole principle there is at least one node $z \in I_{|y-x|}^*(x) \setminus S = I_{|y-x|}(y) \setminus S$ such that z can be reached from x and y can be reached from z in $G-S$.

Claim 4.5 *Let $G=(V=[n],E)$ be a δ-local expander, let $x \in [n]$ be a γ-good node under $S \subseteq [n]$ and let $r>0$ be given. If $\delta < \gamma/2$ then all but $2\delta r$ of the nodes in $I_r^*(x)\backslash S$ are reachable from x in $G-S$. Similarly, x can be reached from all but $2\delta r$ of the nodes in $I_r(x)\backslash S$. In particular, if $\delta < 1/4$ then more than half of the nodes in $I_r^*(x)\backslash S$ (resp. in $I_r(x)\backslash S$) are reachable from x (resp. x is reachable from) in $G-S$.*

Proof. We prove by induction that (1) if $r = 2^k\delta^{-1}$ for some integer k then all but δr of the nodes in $I_r^*(x)\backslash S$ are reachable from x and, (2) if $2^{k-1} < r < 2^k\delta^{-1}$ then then all but $2\delta r$ of the nodes in $I_r^*(x)\backslash S$ are reachable from x. For the base cases we observe that if $r \leq \delta^{-1}$ then, by definition of a δ-local expander, x is directly connected to all nodes in $I_r^*(x)$ so all nodes in $I_r(x)\backslash S$ are reachable.

Now suppose that claims (1) and (2) holds for each $r' \leq r = 2^k\delta^{-1}$. Then we show that the claim holds for each $r < r' \leq 2r = 2^{k+1}\delta^{-1}$. In particular, let $A \subseteq I_r^*(x)\backslash S$ denote the set of nodes in $I_r^*(x)\backslash S$ that are reachable from x via a directed path in $G-S$ and let $B \subseteq I_{r'-r}^*(x+r)\backslash S$ be the set of all nodes in $I_{r'-r}^*(x+r)\backslash S$ that are *not reachable* from x in $G-S$. Clearly, there are no directed edges from A to B in G and by induction we have $|A| \geq |I_r^*(x)\backslash S| - \delta r \geq r(\gamma-\delta) > \delta r$. Thus, by δ-local expansion $|B| \leq r\delta$. Since, $|I_r^*(x)\backslash(S\cup A)| \leq \delta r$ at most $|I_{r'}^*(x)\backslash(S\cup A)| \leq |B| + \delta r \leq 2\delta r \leq 2\delta r'$ nodes in $I_{2r}^*(x)\backslash S$ are not reachable from x in $G-S$. Since, $r' > r$ the number of unreachable nodes is at most $2\delta r \leq 2\delta r'$, and if $r' = 2r$ then the number of unreachable nodes is at most $2\delta r = \delta r'$.

A similar argument shows that x can be reached from all but $2\delta r$ of the nodes in $I_r(x)\backslash S$ in the graph $G-S$. □

Proof of Lemma 4.3. By Claim 4.5 for each r we can reach $|I_r^*(x)\backslash S| - \delta r = |I_r^*(x)\backslash S|\left(1-\delta\frac{|I_r^*(x)|}{|I_r^*(x)\backslash S|}\right) \geq |I_r^*(x)\backslash S|\left(1-\frac{\delta}{\gamma}\right) > \frac{1}{2}|I_r^*(x)\backslash S|$ of the nodes in $I_r^*(x)\backslash S$ from the node x in $G-S$. Similarly, we can reach y from more than $\frac{1}{2}|I_r(x)\backslash S|$ of the nodes in $I_r(y)\backslash S$. Thus, by the pigeonhole principle we can find at least one node $z \in I_{|y-x|}^*(x) \setminus S = I_{|y-x|}(y) \setminus S$ such that z can be reached from x and y can be reached from z in $G-S$. □

Lemma 4.4 shows that almost all of the nodes in $G-S$ are γ-good. The proof is again similar in spirit to an argument of [EGS75]. In particular, [EGS75] constructed a superset T of the set of all γ-bad nodes and then bound the size of this superset T. However, they only prove that $BAD \subset T \subseteq F \cup B$ where $|F|,|B| \leq |S|/(1-\gamma)$. Thus, we have $|BAD| \leq |T| \leq 2|S|/(1-\gamma)$. Unfortunately, this bound is not sufficient for our purposes. In particular, if $|S| = n/2$ then this bound does not rule out the possibility that $|BAD| = n$ so that none of the remaining nodes are good. Instead of bounding the size of the superset T directly we instead bound the size of the set $T \setminus S$ observing that $|BAD| \leq |T| \leq |S| + |T \setminus S|$. In particular, we can show that $|T \setminus S| \leq \frac{2\gamma|S|}{1-\gamma}$. We then have $|GOOD| \geq n - |T| = n - |S| - |T\backslash S| \geq n - |S| - \frac{2\gamma|S|}{1-\gamma}$.

Proof of Lemma 4.4. We say that a γ-bad node x has a forward (resp. backwards) witness r if $|I_r^*(x)\backslash S| > \gamma r$. Let x_1^*, r_1^* be the lexicographically first γ-bad node with a forward witness. Once $x_1^*, r_1^*, \ldots, x_k^*, r_k^*$ have been define let x_{k+1}^* be the lexicographically least γ-bad node such that $x_{k+1}^* > x_k^* + r_k^*$ and x_{k+1}^* has a forward witness r_{k+1}^* (if such a node exists). Let $x_1^*, r_1^*, \ldots, x_k^*, r_{k*}^*$ denote the complete sequence, and similarly define a maximal sequence $x_1, r_1, \ldots, x_k, r_k$ of γ-bad nodes with backwards witnesses such that $x_i - r_i > x_{i+1}$ for each i.

Let

$$F = \bigcup_{i=1}^{k^*} I^*_{r^*_i}\left(x^*_i\right) \quad , \text{and} \quad B = \bigcup_{i=1}^{k} I_{r_i}\left(x_i\right)$$

Note that for each $i \leq k^*$ we have $\left|I^*_{r^*_i}\left(x^*_i\right) \backslash S\right| \leq \gamma r$. Similarly, for each $i \leq k$ we have $\left|I_{r_i}\left(x_i\right) \backslash S\right| \leq \gamma r$. Because the sets $I^*_{r^*_i}\left(x^*_i\right)$ are all disjoint (by construction) we have

$$|F \backslash S| \leq \gamma \sum_{i=1}^{k^*} r^*_i = \gamma |F| \ .$$

Similarly, $|B\backslash S| \leq \gamma |B|$. We also note that at least $(1-\gamma)|F|$ of the nodes in $|F|$ are in $|S|$. Thus, $|F|(1-\gamma) \leq |S|$ and similarly $|B|(1-\gamma) \leq |S|$. We conclude that $|F\backslash S| \leq \frac{\gamma|S|}{1-\gamma}$ and that $|B\backslash S| \leq \frac{\gamma|S|}{1-\gamma}$.

To finish the proof let $T = F \cup B = S \cup (F\backslash S) \cup (B \backslash S)$. Clearly, T is a superset of all γ-bad nodes. Thus, at least $n - |T| \geq n - |S|\left(1 + \frac{2\gamma}{1-\gamma}\right) = n - |S|\frac{1+\gamma}{1-\gamma}$ nodes are good.

We also remark that Lemma 4.2 can be modified to yield a family of DAGs $\{G_n\}_{n=1}^{\infty}$ with $\mathsf{indeg}(G_n) = O\left(\log n \log^* n\right)$ such that G_n is a δ_n local expander for some sequence $\{\delta_n\}_{n=1}^{\infty}$ converging to 0. We can define a sequence $\{\gamma_n\}_{n=1}^{\infty}$ such that $\frac{1+\gamma_n}{1-\gamma_n}$ converges to 1 and $2\gamma_n > \delta_n$ for each n. Lemma 4.2 and Lemma 4.4 then imply that each G_n is $\left(e, n - e\frac{1+\gamma_n}{1-\gamma_n}\right)$-depth robust for any $e \leq n$.

4.1 Additional Applications of Extremely Depth Robust Graphs

We now discuss additional applications of Theorem 4.1.

4.1.1 Application 0: Proofs of Sequential Work

As we previously noted Mahmoody et al. [MMV13] used extremely depth-robust graphs to construct efficient Proofs-Of-Sequential Work. In a proof of sequential work a prover wants to convince a verifier that he computed a hash chain of length n involving the input value x without requiring the verifier to recompute the entire hash chain. Mahmoody et al. [MMV13] accomplish this by requiring the prover computes labels $L_1, \ldots, L_n$ by "pebbling" an extremely depth-robust DAG G_n e.g., $L_{i+1} = H\left(x \| L_{v_1} \| \ldots \| L_{v_\delta}\right)$ where $\{v_1, \ldots, v_\delta\} = \mathsf{parents}(i+1)$ and H is a random oracle. The prover then commits to the labels $L_1, \ldots, L_n$ using a Merkle Tree and sends the root of the tree to the verifier who can audit randomly chosen labels e.g., the verifier audits label L_{i+1} by asking the prover to reveal the values L_{i+1} and L_v for each $v \in \mathsf{parents}(i+1)$. If the DAG is extremely-depth robust then either a (possibly cheating) prover make at least $(1-\epsilon)n$ sequential queries to the random oracle, or the the prover will fail to convince the verifier with high probability [MMV13].

We note that the parameter $\delta = \mathsf{indeg}(G_n)$ is crucial to the efficiency of the Proofs-Of-Sequential Work protocol since each audit challenge requires the prover to reveal $\delta + 1$ labels in the Merkle tree. The DAG G_n from [MMV13] has $\mathsf{indeg}(G_n) = O\left(\log^2 n \cdot \mathsf{polylog}\left(\log n\right)\right)$ while our DAG G_n from Theorem 4.1 has maximum indegree $\mathsf{indeg}(G_n) = O\left(\log n\right)$. Thus, we can improve the communication complexity of the Proofs-Of-Sequential Work protocol by a factor of $\Omega(\log n \cdot \mathsf{polylog} \log n)$.

4.1.2 Application 1: Graphs with Maximum Cumulative Cost

We now show that our family of extreme depth-robust DAGs has the highest possible cumulative pebbling cost even in terms of the *constant* factors. In particular, for any constant $\eta > 0$ the family $\{G^{\eta}_n\}_{n=1}^{\infty}$ of DAGs from Theorem 4.1 has $\Pi^{\parallel}_{cc}(G_n) \geq \frac{n^2(1-\eta)}{2}$ and $\mathsf{indeg}(G_n) = O(\log n)$. By comparison, $\Pi^{\parallel}_{cc}(G_n) \leq \frac{n^2+n}{2}$ for *any* DAG $G \in \mathbb{G}_n$ — even if G is the complete DAG.

Previously, Alwen et al. [ABP17] showed that any (e,d)-depth robust DAG G has $\Pi^{\parallel}_{cc}(G) > ed$ which implies that their is a family of DAG G_n with $\Pi^{\parallel}_{cc}(G_n) = \Omega\left(n^2\right)$ [EGS75]. We stress that we need new techniques to prove Theorem 4.6. Even if a DAG $G \in \mathbb{G}_n$ were $(e, n-e)$-depth robust for every $e \geq 0$ (the only DAG actually satisfying this property is the compete DAG K_n) [ABP17] only implies that $\Pi_{cc}^{\parallel}(G_n) \geq \max_{e\geq 0} e(n-e) = n^2/4$. Our basic insight is that at time t_i, the first time a pebble is placed on node i in G_n^{ϵ}, the node $i+\gamma i$ is γ-good and is therefore reachable via an undirected path from all of the other γ-good nodes in $[i]$. If we have $|P_{t_i}| < (1-\eta/2)\, i$ then we can show that at least $\Omega(\eta i)$ of the nodes in $[i]$ are γ-good. We can also show that these γ-good nodes form a depth robust subset and will cost $\Omega\left((\eta-\epsilon)^2 i^2\right)$ to repebble them by [ABP17]. Since, we would need to pay this cost by time $t_{i+\gamma i}$ it is less expensive to simply ensure that $|P_{t_i}| > (1-\eta/2)\, i$. We refer an interested reader to Appendix A for a complete proof.

Theorem 4.6 *For any constant $0 < \eta < 1$ the family $\{G_n^{\eta}\}_{n=1}^{\infty}$ of DAGs from Theorem 4.1 has $\mathsf{indeg}(G_n) = O(\log n)$ and $\Pi_{cc}^{\parallel}(G_n) \geq \frac{n^2(1-\eta)}{2}$.*

4.1.3 Cumulative Space in Parallel-Black Sequential-White Pebblings

The black-white pebble game [CS76] was introduced to model nondeterministic computations. White pebbles correspond to nondeterministic guesses and can be placed on any vertex at any time, but these pebble can only be removed when (e.g., when we can verify the correctness of this guess). Formally, black white-pebbling configuration $P_i = \left(P_i^W, P_i^B\right)$ of a DAG $G = ([n], E)$ consists of two subsets $P_i^W, P_i^B \subseteq [n]$ where P_i^B (resp. P_i^W) denotes the set of nodes in G with black (resp. white) pebbles on them at time i. For a legal parallel-black sequential-white pebbling $P = (P_0, \ldots, P_t) \in \mathcal{P}_G^{BW}$ we require that we start with no pebbles on the graph i.e., $P_0 = (\emptyset, \emptyset)$ and that all white pebbles are removed by the end i.e., $P_t^W = \emptyset$ so that we verify the correctness of every nondeterministic guess before terminating. If we place a black pebble on a node v during round $i+1$ then we require that all of v's parents have a pebble (either black or white) on them during round i i.e., $\mathsf{parents}\left(P_{i+1}^B \setminus P_i^B\right) \subseteq P_i^B \cup P_i^W$. In the Parallel-Black Sequential-White model we require that at most one new white pebble is placed on the DAG in every round i.e., $\left|P_i^W \setminus P_{i-1}^W\right| \leq 1$ while no such restrict applies for black pebbles. See Definition A.2 in Appendix A for a more formal definition of the parallel-black sequential white pebbling game.

We can use our construction of a family of extremely depth-robust DAG $\{G_n\}_{n=1}^{\infty}$ to establish new upper and lower bounds for

Alwen et al. [AdRNV17] previously showed that in the parallel-black sequential white pebbling model an (e,d)-depth-robust DAG G requires cumulative space at least $\Pi_{cc}^{BW}(G) \doteq \min_{P\in\mathcal{P}_G^{BW}} \sum_{i=1}^{t} \left|P_i^B \cup P_i^W\right| = \Omega\left(e\sqrt{d}\right)$ or at least $\geq ed$ in the sequential black-white pebbling game. In this section we show that any (e,d)-reducible DAG admits a parallel-black sequential white pebbling with cumulative space at most $O(e^2 + dn)$ which implies that any DAG with constant indegree admits a parallel-black sequential white pebbling with cumulative space at most $O(\frac{n^2\log^2\log n}{\log^2 n})$ since any DAG is $(n\log\log n/\log n, n/\log^2 n)$-reducible. We also show that this bound is essentially tight (up to $\log\log n$ factors) using our construction of extremely depth-robust DAGs. In particular, we can find a family of DAGs $\{G_n\}_{n=1}^{\infty}$ with $\mathsf{indeg}(G_n) = 2$ such that any parallel-black sequential white pebbling has cumulative space at least $\Omega(\frac{n^2}{\log^2 n})$. To show this we start by showing that any parallel-black sequential white pebbling of an extremely depth-robust DAG G, with $\mathsf{indeg}(G) = O(\log n)$, has cumulative space at least $\Omega(n^2)$. We use Lemma 2.2 to reduce the indegree of the DAG and obtain a DAG G' with $n' = O(n\log n)$ nodes and $\mathsf{indeg}(G) = 2$, such that any parallel-black sequential white pebbling of G' has cumulative space at least $\Omega(\frac{n^2}{\log^2 n})$.

To the best of our knowledge no general upper bound on cumulative space complexity for parallel-black sequential-white pebblings was known prior to our work other than the parallel black-pebbling attacks of Alwen and Blocki [AB16]. This attack, which doesn't even use the white pebbles, yields an upper bound of $O(ne + n\sqrt{nd})$ for (e,d)-reducible DAGs and $O(n^2\log\log n/\log n)$ in general. One could also consider a "parallel-white parallel-black" pebbling model in which we are allowed to place as many white pebbles as he would like in each round. However, this model admits a trivial pebbling. In particular, we could place white pebbles on every node during the first round and remove all of these pebbles in the next round e.g., $P_1 = (\emptyset, V)$ and $P_2 = (\emptyset, \emptyset)$. Thus, any DAG has cumulative space complexity $\theta(n)$ in the "parallel-white parallel-black" pebbling model.

Theorem 4.7 shows that (e,d)-reducible DAG admits a parallel-black sequential white pebbling with cumulative space at most $O(e^2 + dn)$. The basic pebbling strategy is reminiscent of the parallel black-pebbling attacks of Alwen and Blocki [AB16]. Given an appropriate depth-reducing set S we use the first $e = |S|$ steps to place white pebbles on all nodes in S. Since $G - S$ has depth at most d we can place black pebbles on the remaining nodes during the next d steps. Finally, once we place pebbles on every node we can legally remove the white pebbles. A formal proof of Theorem 4.7 can be found in Appendix A.

Theorem 4.7 *Let $G = (V, E)$ be (e,d)-reducible then $\Pi_{cc}^{BW}(G) \leq \frac{e(e+1)}{2} + dn$. In particular, for any DAG G with $\mathsf{indeg}(G) = O(1)$ we have $\Pi_{cc}^{BW}(G) = O\left(\left(\frac{n \log\log n}{\log n}\right)^2\right)$.*

Theorem 4.8 shows that our upper bound is essentially tight. In a nut-shell their lower bound was based on the observation that for any integers i, d the DAG $G - \bigcup_j P_{i+jd}$ has depth at most d since any remaining path must have been pebbled completely in time d— if G is (e,d)-depth robust this implies that $\left|\bigcup_j P_{i+jd}\right| \geq e$. The key difficulty in adapting this argument to the parallel-black sequential white pebbling model is that it is actually possible to pebble a path of length d in $O(\sqrt{d})$ steps by placing white pebbles on every interval of length $\sqrt{d}$. This is precisely why Alwen et al. [AdRNV17] were only able to establish the lower bound $\Omega(e\sqrt{d})$ for the cumulative space complexity of (e,d)-depth robust DAGs — observe that we always have $e\sqrt{d} \leq n^{1.5}$ since $e + d \leq n$ for any DAG G. We overcome this key challenge by using extremely depth-robust DAGs.

In particular, we exploit the fact that extremely depth-robust DAGs are "recursively" depth-robust. For example, if G is (e,d)-depth robust for any $e + d \leq (1-\epsilon)n$ then the DAG $G - S$ is (e,d)-depth robust for any $e + d \leq (n - |S|) - \epsilon n$. Since $G - S$ is still sufficiently depth-robust we can then show that for some node $x \in V(G-S)$ any (possibly incomplete) pebbling $P = (P_0, P_1, \ldots, P_t)$ of $G - S$ with $P_0 = P_t = (\emptyset, \emptyset)$ either (1) requires $t = \Omega(n)$ steps, or (2) fails to place a pebble on x i.e. $x \notin \bigcup_{r=0}^{t}\left(P_0^W \cup P_r^B\right)$. By Theorem 4.1 it then follows that there is a family of DAGs $\{G_n\}_{n=1}^{\infty}$ with $\mathsf{indeg}(G_n) = O(\log n)$ and $\Pi_{cc}^{BW}(G) = \Omega(n^2)$. If apply indegree reduction Lemma 2.2 to G_n we obain a DAG G'_n with $\mathsf{indeg}(G'_n) = 2$ and $O(n \log n)$ nodes. A similar argument shows that $\Pi_{cc}^{BW}(G) = \Omega(n^2/\log^2 n)$. A formal proof of Theorem 4.8 can be found in Appendix A.

Theorem 4.8 *Let $G = (V = [n], E \supset \{(i, i+1) : i < n\})$ be (e,d)-depth-robust for any $e + d \leq (1-\epsilon)n$ then $\Pi_{cc}^{BW}(G) \geq (1/16 - \epsilon/2)\, n^2$. Furthermore, if $G' = ([2n\delta], E')$ is the indegree reduced version of G from Lemma 2.2 then $\Pi_{cc}^{BW}(G') \geq (1/16 - \epsilon/2)\, n^2$. In particular, there is a family of DAGs $\{G_n\}_{n=1}^{\infty}$ with $\mathsf{indeg}(G_n) = O(\log n)$ and $\Pi_{cc}^{BW}(G) = \Omega(n^2)$, and a separate family of DAGs $\{H_n\}_{n=1}^{\infty}$ with $\mathsf{indeg}(H_n) = 2$ and $\Pi_{cc}^{BW}(H_n) = \Omega\left(\frac{n^2}{\log^2 n}\right)$.*

5 A Pebbling Reduction for Sustained Space Complexity

As an application of the pebbling results on sustained space in this section we construct a new type of moderately hard function (MoHF) in the parallel random oracle model pROM. In slightly more detail, we first fix the computational model and define a particular notion of moderatly hard function called sustained memory-hard functions (SMHF). We do this using the framework of [AT17] so, beyond the applications to password based cryptography, the results in [AT17] for building provably secure cryptographic applications on top of any MoHF can be immediatly applied to SMHFs. In particular this results in a proof-of-work and non-interactive proof-of-work where "work" intuitively means having performed some computation entailing sufficient sustained memory. Finally we prove a "pebbling reduction" for SMHFs; that is we show how to bound the parameters describing the sustained memory complexity of a family of SMHFs in terms of the sustained space of their underlying graphs.[3]

[3]Effectively this does for SMHFs what [AT17] did for MHFs.

5.1 Defining Sustained Memory Hard Functions

We very briefly sketch the most important parts of the MoHF framework of [AT17] which is, in turn, a generalization of the indifferentiability framework of [MRH04].

We begin with the following definition which describes a family of functions that depend on a (random) oracle.

Definition 5.1 (Oracle functions) *For (implicit) oracle set* $\mathbb{H}$*, an* oracle function $f^{(\cdot)}$ (with domain D and range R)*, denoted* $f^{(\cdot)} : D \to R$*, is a set of functions indexed by oracles* $h \in \mathbb{H}$ *where each* f^h *maps* $D \to R$*.*

Put simply, an MoHF is a pair consisting of an oracle family $f^{(\cdot)}$ and an honest algorithm $\mathcal{N}$ for evaluating functions in the family using access to a random oracle. Such a pair is secure relative to some computational model M if no adversary $\mathcal{A}$ with a computational device adhering to M (denoted $\mathcal{A} \in M$) can produce output which couldn't be produced simply by called $f^{(h)}$ a limited number of times (where h is a uniform choice of oracle from $\mathbb{H}$). It is asumed that algorithm $\mathcal{N}$ is computable by devices in some (possibly different) computational model $\bar{M}$ when given sufficent computational resources. Usually M is strictly more powerful than $\bar{M}$ reflecting the assumption that an adversary could have a more powerful class of device than the honest party. For example, in this work we will let model $\bar{M}$ contain only sequential devices (say Turing machines which make one call to the random oracle at a time) while M will also include parallel devices.

In this work, both the computational models M and $\bar{M}$ are parametrized by the same space $\mathbb{P}$. For each model, the choice of parameters fixes upperbounds on the power of devices captured by that model; that is on the computational resources available to the permitted devices. For example M_a could be all Turing machines making at most a queries to the random oracle. The security of a given moderatly hard function is parameterized by two functions α and β mapping the parameter space for M to positive integers. Intuitively these functions are used to provide the following two properties.

Completeness: To ensure the construction is even useable we require that $\mathcal{N}$ is (computable by a device) in model M_a and that $\mathcal{N}$ can evaluate $f^{(h)}$ (when given access to h) on at least $\alpha(a)$ distinct inputs.

Security: To capture how bounds on the resources of an adversary $\mathcal{A}$ limit the ability of $\mathcal{A}$ to evalute the MoHF we require that the output of $\mathcal{A}$ when running on a device in model M_b (and having access to the random oracle) can be reproduced by some simulator σ using at most $\beta(b)$ oracle calls to $f^{(h)}$ (for uniform randomly sampled $h \leftarrow \mathbb{H}$.

To help build provably secure applications on top of MoHFs the framework makes use of a destinguisher $\mathcal{D}$ (similar to the environment in the Universal Composability[Can01] family of models or, more accurately, to the destinguisher in the indifferentiability framework). The job of $\mathcal{D}$ is to (try to) tell a *real world* interaction with $\mathcal{N}$ and the adversary $\mathcal{A}$ apart from an *ideal world* interaction with $f^{(h)}$ (in place of $\mathcal{N}$) and a simulator (in place of the adversary). Intuitivelly, $\mathcal{D}$'s access to $\mathcal{N}$ captures whatever $\mathcal{D}$ could hope to learn by interacting with an arbitrary application making use of the MoHF. The definition then ensures that even leveraging such information the adversary $\mathcal{A}$ can not produce anything that could not be simulated (by simulator σ) to $\mathcal{D}$ using nothing more than a few calls to $f^{(h)}$.

As in the above description we have ommited several details of the framework we will also use a somewhat simplified notation. We denote the above described real world execution with the pair $(\mathcal{N}, \mathcal{A})$ and an ideal world execution where $\mathcal{D}$ is permited $c \in \mathbb{N}$ calls to $f^{(\cdot)}$ and simulator σ is permited $d \in \mathbb{N}$ calls to $f^{(h)}$ with the pair $(f^{(\cdot)}, \sigma)_{c,d}$. To denote the statement that no $\mathcal{D}$ can tell an interaction with $(\mathcal{N}, \mathcal{A})$ apart one with $(f^{(\cdot)}, \sigma)_{c,d}$ with more than probability ϵ we write $(\mathcal{N}, \mathcal{A}) \approx_\epsilon (f^{(\cdot)}, \sigma)_{c,d}$.

Finally, to accomadate honest parties with varying amounts of resources we equip the MoHF with a hardness parameter $n \in \mathbb{N}$. The following is the formal security definition of a MoHF. Particular types of MoHF (such as the one we define bellow for sustained memory complexity) differ in the precise notion of computational model they consider. For further intution, a much more detailed exposition of the framework and how the following definition can be used to prove security for applications we refer to [AT17].

Definition 5.2 (MoHF security) *Let M and $\bar{M}$ be computational models with bounded resources parametrized by $\mathbb{P}$. For each $n \in \mathbb{N}$, let $f_n^{(\cdot)}$ be an oracle function and $\mathcal{N}(n,\cdot)$ be an algorithm (computable by some device in $\bar{M}$) for evaluating $f_n^{(\cdot)}$. Let $\alpha, \beta : \mathbb{P} \times \mathbb{N} \to \mathbb{N}$, and let $\epsilon : \mathbb{P} \times \mathbb{P} \times \mathbb{N} \to \mathbb{R}_{\geq 0}$. Then, $(f_n^{(\cdot)}, \mathcal{N}_n)_{n\in\mathbb{N}}$ is a* $(\alpha, \beta, \epsilon)$-secure moderately hard function family (for model M) *if*

$$\forall n \in \mathbb{N}, \mathbf{r} \in \mathbb{P}, \mathcal{A} \in M_{\mathbf{r}} \; \exists \sigma \; \forall \mathbf{l} \in \mathbb{P} : \quad (\mathcal{N}(n,\cdot), \mathcal{A}) \; \approx_{\epsilon(\mathbf{l},\mathbf{r},n)} \; (f_n^{(\cdot)}, \sigma)_{\alpha(\mathbf{l},n),\beta(\mathbf{r},n)} \;, \tag{3}$$

The function family is asymptotically secure *if $\epsilon(\mathbf{l}, \mathbf{r}, \cdot)$ is a negligible function in the third parameter for all values of $\mathbf{r}, \mathbf{l} \in \mathbb{P}$.*

Sustained Space Constrained Computation. Next we define the honest and adversarial computational models for which we prove the pebbling reduction. In particular we first recall (a simplified version of) the pROM of [AT17]. Next we define a notion of sustained memory in that model naturally mirroring the notion of sustained space for pebbling. Thus we can parametrize the pROM by memory threshold s and time t to capture all devices in the pROM with no more sustained memory complexity then given by the choice of those parameters.

In more detail, we consider a resource-bounded computational device $\mathcal{S}$. Let $w \in \mathbb{N}$. Upon startup, $\mathcal{S}^{w\text{-PROM}}$ samples a fresh random oracle $h \leftarrow_{\$} \mathbb{H}_w$ with range $\{0,1\}^w$. Now $\mathcal{S}^{w\text{-PROM}}$ accepts as input a pROM algorithm $\mathcal{A}$ which is an oracle algorithm with the following behavior.

A *state* is a pair $(\tau, \mathbf{s})$ where *data* τ is a string and $\mathbf{s}$ is a tuple of strings. The output of step i of algorithm $\mathcal{A}$ is an *output state* $\bar{\sigma}_i = (\tau_i, \mathbf{q}_i)$ where $\mathbf{q}_i = [q_i^1, \ldots, q_i^{z_i}]$ is a tuple of *queries* to h. As input to step $i+1$, algorithm $\mathcal{A}$ is given the corresponding *input state* $\sigma_i = (\tau_i, h(\mathbf{q}_i))$, where $h(\mathbf{q}_i) = [h(q_i^1), \ldots, h(q_i^{z_i})]$ is the tuple of *responses* from h to the queries $\mathbf{q}_i$. In particular, for a given h and random coins of $\mathcal{A}$, the input state σ_{i+1} is a function of the input state σ_i. The initial state σ_0 is empty and the input x_{in} to the computation is given a special input in step 1.

For a given execution of a pROM, we are interested in the following new complexity measure parametrized by an integer $s \geq 0$. We call an element of $\{0,1\}^s$ a *block*. Moreover, we denote the bit-length of a string r by $|r|$. The *length* of a state $\sigma = (\tau, \mathbf{s})$ with $\mathbf{s} = (s^1, s^2, \ldots, s^y)$ is $|\sigma| = |\tau| + \sum_{i\in[y]} |s^i|$. For a given state σ let $b(\sigma) = \lfloor |\sigma|/s \rfloor$ be the number of "blocks in σ". Intuitively, the s-sustained memory complexity (s-SMC) of an execution is the sum of the number of blocks in each state. More precisely, consider an execution of algorithm $\mathcal{A}$ on input x_{in} using coins \$ with oracle h resulting in $z \in \mathbb{Z}_{\geq 0}$ input states $\sigma_1, \ldots, \sigma_z$, where $\sigma_i = (\tau_i, \mathbf{s}_i)$ and $\mathbf{s}_i = (s_i^1, s_i^2, \ldots, s_i^{y_j})$. Then the for integer $s \geq 0$ the *s-sustained memory complexity* (s-SMC) of the execution is

$$s\text{-}\mathsf{smc}(\mathcal{A}^h(x_{\mathsf{in}}; \$)) = \sum_{i\in[z]} b(\sigma_i) \;,$$

while the *total number of RO calls* is $\sum_{i\in[z]} y_j$. More generally, the s-SMC (and total number of RO calls) of several executions is the sum of the s-sMC (and total RO calls) of the individual executions.

We can now describe the resource constraints imposed by $\mathcal{S}^{w\text{-PROM}}$ on the pROM algorithms it executes. To quantify the constraints, $\mathcal{S}^{w\text{-PROM}}$ is parametrized by element from $\mathbb{P}^{\text{PROM}} = \mathbb{N}^3$ which describe the limites on an execution of algorithm $\mathcal{A}$. In particular, for parameters $(q, s, t) \in \mathbb{P}^{\text{PROM}}$, algorithm $\mathcal{A}$ is allowed to make a total of q RO calls and have s-SMC at most t (summed across all invocations of $\mathcal{A}$ in any given experiment).

As usual for moderately hard functions, to ensure that the honest algorithm can be run on realistic devices, we restrict the honest algorithm $\mathcal{N}$ for evaluating the SMHF to be a *sequential* algorithms. That is, $\mathcal{N}$ can make only a single call to h per step. Technically, in any execution, for any step j it must be that $y_j \leq 1$. No such restriction is placed on the adversarial algorithm reflecting the power (potentially) available to such a highly parallel device as an ASIC. In symbols we denote the sequential version of the pROM, which we refer to as the sequential ROM (sROM) by $\mathcal{S}^{w\text{-SROM}}$.

We can now (somewhat) formally define of a sustained memory-hard function for the pROM. The definition is a particular instance of and moderately hard function (c.f. Definition 5.2).

Definition 5.3 (Sustained Memory-Hard Function) *For each $n \in \mathbb{N}$, let $f_n^{(\cdot)}$ be an oracle function and $\mathcal{N}_n$ be an sROM algorithm for computing $f^{(\cdot)}$. Consider the function families:*

$$\alpha = \{\alpha_w : \mathbb{P}^{\text{PROM}} \times \mathbb{N} \to \mathbb{N}\}_{w \in \mathbb{N}} , \quad \beta = \{\beta_w : \mathbb{P}^{\text{PROM}} \times \mathbb{N} \to \mathbb{N}\}_{w \in \mathbb{N}} ,$$

$$\epsilon = \{\epsilon_w : \mathbb{P}^{\text{PROM}} \times \mathbb{P}^{\text{PROM}} \times \mathbb{N} \to \mathbb{N}\}_{w \in \mathbb{N}} .$$

*Then $F = (f_n^{(\cdot)}, \mathcal{N}_n)_{n \in \mathbb{N}}$ is called an $(\alpha, \beta, \epsilon)$-*sustained memory-hard function *(SMHF) if $\forall w \in \mathbb{N}$ F is an $(\alpha_w, \beta_w, \epsilon_w)$-secure moderately hard function family for $\mathcal{S}^{w\text{-PROM}}$.*

5.2 The Construction

In this work $f^{(\cdot)}$ will be a graph function [AS15] (also sometimes called "hash graph"). The following definition is taken from [AT17]. A graph function depends on an oracle $h \in \mathbb{H}_w$ mapping bit strings to bit strings. We also assume the existance of an implicit prefix-free encoding such that h is evaluated on unique strings. Inputs to h are given as distinct tuples of strings (or even tuples of tuples of strings). For example, we assume that $h(0, 00)$, $h(00, 0)$, and $h((0, 0), 0)$ all denote distinct inputs to h.

Definition 5.4 (Graph function) *Let function $h : \{0,1\}^* \to \{0,1\}^w \in \mathbb{H}_w$ and DAG $G = (V, E)$ have source nodes $\{v_1^{\mathsf{in}}, \ldots, v_a^{\mathsf{in}}\}$ and sink nodes $(v_1^{\mathsf{out}}, \ldots, v_z^{\mathsf{out}})$. Then, for inputs $\mathbf{x} = (x_1, \ldots, x_a) \in (\{0,1\}^*)^{\times a}$, the $(h, \mathbf{x})$-*labeling *of G is a mapping $\mathsf{lab} : V \to \{0,1\}^w$ defined recursively to be:*

$$\forall v \in V \quad \mathsf{lab}(v) := \begin{cases} h(\mathbf{x}, v, x_j)) & : v = v_j^{\mathsf{in}} \\ h(\mathbf{x}, v, \mathsf{lab}(v_1), \ldots, \mathsf{lab}(v_d))) & : else \end{cases}$$

where $\{v_1, \ldots, v_d\}$ are the parents of v arranged in lexicographic order.

The graph function (of G and $\mathbb{H}_w$) *is the oracle function*

$$f_G : (\{0,1\}^*)^{\times a} \to (\{0,1\}^w)^{\times z} ,$$

which maps $\mathbf{x} \mapsto (\mathsf{lab}(v_1^{\mathsf{out}}), \ldots, \mathsf{lab}(v_z^{\mathsf{out}}))$ where lab is the $(h, \mathbf{x})$-labeling of G.

Given a graph function we need an honest (sequential) algorithm for computing it in the pROM. For this we use the same algorithm as already used in [AT17]. The honest oracle algorithm $\mathcal{N}_G$ for graph function f_G computes one label of G at a time in topological order appending the result to its state. If G has $|V| = n$ nodes then $\mathcal{N}_G$ will terminate in n steps making at most 1 call to h per step, for a total of n calls, and will never store more than $n * w$ bits in the data portion of its state. In particular for all inputs $\mathbf{x}$, oracles h (and coins \$) we have that for any $s \in [n]$ if the range of h is in $\{0,1\}^w$ then algorithm $\mathcal{N}$ has sw-SMC of $n - s$.

Recall that we would like to set $\alpha_w : \mathbb{P}^{\text{PROM}} \to \mathbb{N}$ such that for any parameters (q, s, t) constraining the honest algorithms resources we are still guaranteed at least $\alpha_w(q, s, t)$ evaluations of f_G by $\mathcal{N}_G$. Given the above honest algorithm we can thus set:

$$\forall (q, s, t) \in \mathbb{P}^{\text{PROM}} \quad \alpha_w(q, s, t) := \begin{cases} 0 & : q < n \\ \min(\lfloor q/n \rfloor, \lfloor t/(n - \lfloor s/w \rfloor \rfloor) & : \text{else} \end{cases}$$

It remains to determine how to set β_w and ϵ_w, which is the focus of the remainder of this section.

5.3 The Pebbling Reduction

We state the main theorem of this section which relates the parameters of an SMHF based on a graph function to the sustained (pebbling) space complexity of the underlying graph.

Theorem 5.5 *[Pebbling reduction] Let $G_n = (V_n, E_n)$ be a DAG of size $|V_n| = n$. Let $F = (f_{G,n}, \mathcal{N}_{G,n})_{n\in\mathbb{N}}$ be the graph functions for G_n and their naïve oracle algorithms. Then, for any $\lambda \geq 0$, F is an $(\alpha, \beta, \epsilon)$-sustained memory-hard function where*

$$\alpha = \{\alpha_w(q, s, t)\}_{w\in\mathbb{N}} \ ,$$

$$\beta = \left\{\beta_w(q, s, t) = \frac{\Pi^{\parallel}_{ss}(G, s)(w - \log q)}{1 + \lambda}\right\}_{w\in\mathbb{N}} \ , \quad \epsilon = \left\{\epsilon_w(q, m) \leq \frac{q}{2^w} + 2^{-\lambda}\right\}_{w\in\mathbb{N}} \ .$$

The technical core of the proof follows that of [AT17] closely. For completeness we briefly sketch the proof in Appendix A.1.

6 Open Questions

We conclude with several open questions for future research. The primary challenge is to provide a practical construction of a DAG G with high sustained space complexity. While we provide a DAG G with asymptotically optimal sustained space complexity, we do not optimize for constant factors. We remark that for practical applications to iMHFs it should be trivial to evaluate the function $\mathsf{parents}_G(v)$ without storing the DAG G in memory explicitly. Toward this end it would be useful to either prove or refute the conjecture that any depth-robustness is sufficient for high sustained space complexity e.g., what is the sustained space complexity of the depth-robust DAGs from [EGS75] or [PTC76]? Another interesting direction would be to relax the notion of sustained space complexity and instead require that for any pebbling $P \in \mathcal{P}^{\parallel}(G)$ either (1) P has large cumulative complexity e.g., n^3, or (2) P has high sustained space complexity. Is it possible to design a dMHF with the property for any evaluation algorithm either has (1) sustained space complexity $\Omega(n)$ for $\Omega(n)$ rounds, or (2) has cumulative memory complexity $\omega(n^2)$?

Acknowledgments

The first and third authors were supported by the European Research Council, ERC consolidator grant (682815 - TOCNeT).

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`.

[ABP17] Joël Alwen, Jeremiah Blocki, and Krzysztof Pietrzak. Depth-robust graphs and their cumulative memory complexity. In *EUROCRYPT*, LNCS, 2017. `https://eprint.iacr.org/2016/875`.

[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.

[AdRNV17] Joël Alwen, Susanna F de Rezende, Jakob Nordström, and Marc Vinyals. Cumulative space in black-white pebbling and resolution. In *8th Innovations in Theoretical Computer Science (ITCS) conference, Berkeley, January 9-11, 2017*, 2017.

[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`.

[AT17] Joël Alwen and Björn Tackmann. Moderately Hard Functions: Definition, Instantiations, and Applications, 2017. `https://eprint.iacr.org/2017/`.

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

[Can01] Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *42nd Annual Symposium on Foundations of Computer Science*, pages 136–145, Las Vegas, Nevada, October 2001. IEEE.

[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.

[Cox16] Bill Cox. Re: [Cfrg] Balloon-Hashing or Argon2i. CFRG Mailinglist, August 2016. `https://www.ietf.org/mail-archive/web/cfrg/current/msg08426.html`.

[CS76] Stephen Cook and Ravi Sethi. Storage requirements for deterministic polynomialtime recognizable languages. *Journal of Computer and System Sciences*, 13(1):25–37, 1976.

[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.

[DKW11] 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 *Advances in Cryptology - CRYPTO 2005: 25th Annual International Cryptology Conference, Santa Barbara, California, USA, August 14-18, 2005, Proceedings*, volume 3621 of *Lecture Notes in Computer Science*, pages 37–54. Springer, 2005.

[EGS75] Paul Erdös, Ronald L. Graham, and Endre Szemerédi. On sparse graphs with dense long paths. Technical report, Stanford, CA, USA, 1975.

[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.

[HPV77] John Hopcroft, Wolfgang Paul, and Leslie Valiant. On time versus space. *J. ACM*, 24(2):332–337, April 1977.

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

[MMV13] Mohammad Mahmoody, Tal Moran, and Salil P. Vadhan. Publicly verifiable proofs of sequential work. In Robert D. Kleinberg, editor, *Innovations in Theoretical Computer Science, ITCS '13, Berkeley, CA, USA, January 9-12, 2013*, pages 373–388. ACM, 2013.

[MRH04] Ueli Maurer, Renato Renner, and Clemens Holenstein. Indifferentiability, impossibility results on reductions, and applications to the random oracle methodology. In *TCC*, volume 2951 of *LNCS*, pages 21–39, 2004.

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

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

[PTC76] Wolfgang J. Paul, Robert Endre Tarjan, and James R. Celoni. Space bounds for a game on graphs. In *Proceedings of the Eighth Annual ACM Symposium on Theory of Computing*, STOC '76, pages 149–160, New York, NY, USA, 1976. ACM.

[RD16] Ling Ren and Srinivas Devadas. Proof of space from stacked expanders. In *TCC 2016-B, Part I*, LNCS, pages 262–285. Springer, Heidelberg, November 2016.

[RD17] Ling Ren and Srinivas Devadas. Bandwidth hard functions for asic resistance. Cryptology ePrint Archive, Report 2017/225, 2017. `http://eprint.iacr.org/2017/225`.

A Missing Proofs

Reminder of Lemma 2.8. *[Indegree Reduction for Parallel Sustained Space]*

$$\forall G \in \mathbb{G}_{n,\delta}, \quad \exists H \in \mathbb{G}_{n',2} \textit{ such that } \forall s \geq 0 \;\; \Pi^{\parallel}_{ss}(H, s/(\delta-1)) = \Pi^{\parallel}_{ss}(G, s) \textit{ where } n' \in [n, \delta n].$$

Proof of Lemma 2.8. To obtain H from G we replace each node v in G with a path of length $\mathsf{indeg}(v)$ and distribute the incoming edges of v along the path. More precicely let $G = (V, E)$ with sinks $S \subseteq V$. For each $v \in V$ let $\delta_v = \mathsf{indeg}(v)$ and $p_{v,i} \in V$ be the i^{th} parent of v (sorted in some arbitrary fixed order). By convention $p_{s,0} = \perp$ for all $s \in S$. We define $H = (V', E')$ as follows. The set of nodes $V' \subseteq V \times [\delta] \cup \{\perp\}$ is

$$V' = \{\langle s, \perp\rangle : s \in S\} \cup \{\langle v, i\rangle : v \in V \setminus S, i \in [\delta_v]\}.$$

The edge set is given by:

$$E' = \{(\langle v, i-1\rangle, \langle v, i\rangle) : v \in V \setminus S, i \in [\delta_v]\} \bigcup \{(\langle u, \delta_u\rangle, \langle v, i\rangle) : (u, v) \in E, u = p_{v,i}\}.$$

Each node of G is replaced by at most δ nodes in H so the size n' of H is $n' \in [n, \delta n]$. Moreover, by construction, no node in H has more than two incoming edges so $H \in \mathbb{G}_{n',2}$ as desired.

Next we map any $P' \in \mathcal{P}^{\parallel}_H$ to a $P \in \mathcal{P}^{\parallel}_G$ and show that $\forall s \geq 0$ we have $\Pi^{\parallel}_{ss}(P', s) \geq \Pi^{\parallel}_{ss}(P, s/(\delta-1))$. In more detail, given $P' = (P'_0, \ldots, P'_z) \in \mathcal{P}^{\parallel}_H$ we define $P = (P_0, \ldots, P_z)$ as follows.

1. For all $i \in [0, z]$ if $\langle v, \delta_v\rangle \in P'_i$ then put v in P_i.
2. Further if $\langle v, j\rangle \in P'_i$ for $j < \delta_v$ then put $(u_1, u_2, \ldots, u_j)$ in to P_i.

Claim A.1 $P' \in \mathcal{P}^{\parallel}_H \implies P \in \mathcal{P}^{\parallel}_G$.

Proof. By assumption $P'_0 = \emptyset$ so $P_0 = \emptyset$. Moreover when a sink $\langle v, \delta_v\rangle \in V'$ of H is pebbled by P' at time i then the sink $v \in V$ of G is pebbled in P'. But any sink of G is mapped to a path in H ending in a sink of H. Thus if all sinks of H are pebbled by P' then so must all sinks of G be pebbled by P. In particular, as by assumption P' is complete so is P.

To prove the claim it remains to show that if P' is a legal pebbling for H then so is P a legal pebbling of G. Suppose, for the sake of contradiction that this is not the case and let $i \in [0, z]$ be the first time a pebble is placed illegally by P and let it be on node $v \in V$. Suppose it was placed due to rule 1. Then it must be that $\langle v, \delta_v\rangle \in P'_i$. Further, as $v \notin P_{i-1}$ it must also be that $\langle v, \delta_v\rangle \notin P'_{i-1}$. By assumption P' is legal so $\mathsf{parents}_H(\langle v, \delta_v\rangle)$ must be pebbled in P_{i-1}. If $\delta_v = \perp$ then v is a source node which contradicts it being pebbled illegally. If $\delta_v = 1$ then there exists node $u = p_{v,1} \in V$ and $\langle u, \delta_u\rangle \in P'_{i-1}$ which, according to rule 1 above implies that $u \in P_{i-1}$. However that too is a contradiction to v being pebbled illegally. If

$\delta_v > 1$ then both $\langle u, \delta_v\rangle$ and $\langle v, \delta_v - 1\rangle$ are in P'_{i-1}. But by rules 1 and 2 then all parents of v are pebbled in P_{i-1} which is again a contradiction to v being pebbled illegally at time i. Thus no node can be illegally pebbled due to rule 1.

Let us suppose instead that v was pebbled illegally due to rule 2 being applied to a pebbled $\langle u, i\rangle$. That is for some $j \in [i]$ we have $v = p_{u,j}$. Since $v \notin_P i-1$ and P' is legal it must be that $j = i$. Moreover, it must be that $\mathsf{parents}_H(\langle u, j\rangle) \in P'_{i-1}$. In particular, then $\langle v, \delta_v, \in\rangle P'_{i-1}$. But then rule 1 implies that $v \in P_{i-1}$ which contradicts v being pebbled illegally by P at time i. $\square$

To complete the proof of the lemma it remains only to relate the threshold complexities of P and P'. Notice that for all $i \in [0, z]$ and any $v \in P_i$ at most $\delta - 1$ new pebbles where added to P_i. Thus we have that $\forall s \geq 0$ it holds that $\Pi^{\parallel}_{ss}(P', s/(\delta-1)) \geq \Pi^{\parallel}_{ss}(P, s)$. $\square$

Reminder of Lemma 4.2. *[EGS75] Let $\delta > 0$ be a fixed constant then there is a family of DAGs $\{G_n^\delta\}_{n=1}^{\infty}$ with $\mathsf{indeg} = O(\log n)$ such that each G_n^δ is a δ-local expander.*

Proof of Lemma 4.2. (sketch) We closely follow the construction/proof of [EGS75]. In particular, we say that a bipartite DAG $T_m^\delta = (V = A \cup B, E)$ with $|A| = |B| = m$ is a δ-expander if for all $X \subseteq A, Y \subseteq B$ such that $|X| \geq \delta m$ and $|Y| \geq \delta m$ we have $E \cap X \times Y \neq \emptyset$ i.e., there is an edge from some node $x \in X$ to some node $y \in Y$. For any constant δ we can find constants c_δ, m_δ such that for all $m > m_\delta$ there exists a T_m^δ expander with $\mathsf{indeg}\left(T_m^\delta\right) \leq c_\delta$ e.g., see the first lemma in [EGS75] [4]. Now following [EGS75] we construct $G_n^\delta = ([n], E_n)$ by repeating the following steps for each $j \in (\lfloor \log_2 m_\delta \rfloor, \lfloor \log_2 m_{\delta/10} \rfloor)$.

1. We partition the nodes $[n]$ into $r = \lceil n/2^j \rceil$ sets $D_{1,j}, \ldots, D_{r,j}$ where $D_i = [i2^j + 1, (i+1)2^j]$.

2. For each $v \leq r$ each $i \in [10]$ such that $v + i \leq r$ we overlay the DAG $T_{2^j}^{\delta/10}$ on top of $D_{v,j}$ and $D_{v+i,j}$ (Edge Case: if $|D_{r,j}| = q \leq 2^j$ then we instead overlay $T_{2^j}^{\delta/10} - \{b_{q+1}, \ldots, b_{2^j}\}$ on top of D_{r-i} and D_r for $i \in [10]$, where $T_{2^j}^{\delta/10} = (V = A \cup B, E)$ and $\{b_{q+1}, \ldots, b_{2^j}\}$ denotes the last $2^j - q$ nodes in B.).

By overlaying these expander graphs we can ensure that for *any* node $v \in [n]$ of G_n^δ and any interval $r \geq m_{\delta/10}$ we have the property that for all $X \subseteq [v, v+r-1] Y \subseteq [v+r, v+2r-1]$ such that $|X| \geq \delta r$ and $|Y| \geq \delta r$ we have $E_n \cap X \times Y \neq \emptyset$ e.g., see [EGS75, Claim 1]. Finally, to ensure local expansion between intervals of the form $[v, v+r-1]$ and $[v+r, v+2r-1]$ with $r < m_{\delta/10}$ we can add all edges of the form $\{(i, i+j) \ : n \geq i+j \wedge j - i \leq \max\{m_{\delta/10}, 4\log n\}\}$. This last step is a modest deviation of [EGS75] since we want to ensure that G_n^δ is a δ-local expander for all $n > 0$ and any constant $\delta > 0$. The graph has $\mathsf{indeg}(G_n) \leq 10 c_\delta \log n + \max\{m_{\delta/10}, 4\log n\} = O(\log n)$.

Reminder of Theorem 4.6. *For any constant $0 < \eta < 1$ the family $\{G_n^\eta\}_{n=1}^{\infty}$ of DAGs from Theorem 4.1 has $\mathsf{indeg}(G_n) = O(\log n)$ and $\Pi^{\parallel}_{cc}(G_n) \geq \frac{n^2(1-\eta)}{2}$.*

Proof of Theorem 4.6. We set $\epsilon = \eta^2/100$ and let G_n^η be the graph G_n^ϵ from the proof of Theorem 4.1. In particular, G_n^η is a $\delta = \epsilon/10$-local expander and we set $\gamma = \epsilon/4$ when we consider γ-good nodes.

Consider a legal pebbling $P \in \mathcal{P}^{\parallel}_{G_n^\eta}$ and let t_i denote the first time that node i is pebbled ($i \in P_{t_i}$, but $i \notin \bigcup_{j<t_i} P_j$). We consider two cases:

Case 1 $|P_{t_i}| \geq (1 - \eta/2)\, i$. Observe that if this held for all i then we immediately have $\sum_{j=1}^{t} |P_i| \geq \sum_{j=1}^{n} |P_{t_i}| \geq (1-\eta/2)\sum_{i=1}^{n} i \geq \frac{n^2(1-\epsilon/2)}{2}$.

Case 2 $P_{t_i} < (1-\eta/2)\, i$. Let $GOOD_i$ denote the set of γ-good nodes in $[i]$. We observe that at least $i - (1-\eta/2) i \frac{1-\gamma}{1+\gamma} \geq i\eta/4$ of the nodes in $[i]$ are γ-good by Lemma 4.4. Furthermore, we note that the subgraph $H_i = G_n^\eta[GOOD_i]$ is $(a\,|Good_i|, (1-a)\,|Good_i| - \epsilon i)$-depth robust for any constants $a > 0$. [5]

[4] In fact, the argument is probabilistic and there is a randomized algorithm which, except with negligibly small probability $\mathsf{negl}(m)$, constructs a δ-expander T_m^δ with m nodes and $\mathsf{indeg}\left(T_m^\delta\right) \leq 2c_\delta$.

[5] To see this observe that if G_n^ϵ is a δ-local expander then $G_n^\epsilon[i]$ is also a δ-local expander. Therefore, Lemma 4.3 and Lemma 4.4 imply that $G_n^\epsilon[i]$ (ai, bi)-depth robust for any $a + b \leq 1 - \epsilon$. Since, H_i is a subgraph of $G_n^\epsilon[i]$ it must be that H_i is

Thus, a result of Alwen et al. [ABP17] gives us $\Pi^{\parallel}_{cc}(H_i) \geq i^2\eta^2/100$ since the DAG H_i is at least $(i\eta/10, i\eta/10)$-depth robust. To see this set $a = 1/2$ and observe that $a|Good_i| \geq i\eta/8$ and that $(1-a)|Good_i| - \epsilon i \geq i\eta/8 - \eta i/100 \geq i\eta/10$. Similarly, we note that at time t_i the node $i + \gamma i$ is γ-good. Thus, by Lemma 4.3 we will have to completely repebble H_i by time $t_{i+\gamma i}$. This means that $\sum_{j=t_i}^{t_{i+\gamma i}} |P_j| \geq \Pi^{\parallel}_{cc}(H_i) \geq i^2\eta^2/100$ and, since $\gamma = \eta^2/400$ we have $i^2\eta^2/100 > 2\gamma i^2 > \sum_{j=i}^{i+\gamma i} j(1-\eta/2)$.

Let x_1 denote the first node $1 \leq x_1 \leq n - \gamma n$ for which $\left|P_{t_{x_1}}\right| < (1-\eta/2)\, i$ and, once $x_1, \ldots, x_k$ have been defined let x_{k+1} denote the first node such that $n - \gamma n > x_{k+1} > \gamma x_k + x_k$ and $\left|P_{t_{x_{k+1}}}\right| < (1-\eta/2)\, i$. Let $x_1, \ldots, x_{k*}$ denote a maximal such sequence and let $F = \bigcup_{j=1}^{k*}[x_j, x_j + \gamma x_j]$. Let $R = [n-\gamma n] \setminus F$. We have $\sum_{j\in R} |P_j| \geq \sum_{j \in R} j(1-\eta/2)$ and we have $\sum_{j \in F} |P_j| \geq \sum_{j\in R} j(1-\eta/2)$. Thus,

$$\sum_{j=1}^{t} |P_i| \geq \sum_{j\in R} |P_j| + \sum_{j \in F} |P_j| \geq \sum_{j=1}^{n-\gamma n} \frac{n^2(1-\eta/2)}{2} \geq \frac{n^2(1-\eta/2)}{2} - \gamma n^2 \geq \frac{n^2(1-\eta)}{2} \ .$$

Definition A.2 (Parallel White 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* black-white pebbling configuration *(of G) consists of two subset $P_i^B, P_i^W \subseteq V$. A legal* parallel *pebbling of T is a sequence $P = (P_0, \ldots, P_t)$ of* black-white pebbling configurations *of G where $P_0 = (\emptyset, \emptyset)$ and which satisfies the following conditions: (1) the last pebbling configuration contains no white pebbles i.e. $P_t = (P_t^B, P_t^W)$ where $P_t^W = \emptyset$, (2) at most one white pebble is placed per step i.e. $\forall i \in [t] \ : \ |P_i^W \setminus P_{i-1}^W| \leq 1$, (3) a white pebble can only be removed from a node if all of its parents were pebbled at the end of the previous step i.e., $\forall i \in [t] \ : \ x \in (P_{i-1}^W \setminus P_i^W) \Rightarrow \mathsf{parents}(x) \subseteq P_{i-1}^W \cup P_{i-1}^B$, (4) a black pebble can only be added if all its parents were pebbled at the end of the end of the previous step i.e., $\forall i \in [t] \ : \ x \in (P_i^B \setminus P_{i-1}^B) \Rightarrow \mathsf{parents}(x) \subseteq P_{i-1}^W \cup P_{i-1}^B$, (5) at some step every node is pebbled (though not necessarily simultaneously) i.e., $\forall x \in T\ \exists z \leq t \ : \ x \in P_z^W \cup P_z^B$. We denote with $\mathcal{P}_G^{BW}$ the set of all parallel-black sequential white pebblings of G. We use $\Pi_{cc}^{BW}(G) = \min_{P \in \mathcal{P}_G^{BW}} \Pi_{cc}^{BW}(P)$ where for $P = (P_0, \ldots, P_t)$ we have $\Pi_{cc}^{BW}(P) = \sum_{i=1}^{t} \left|P_i^B \cup P_i^W\right|$.*

Reminder of Theorem 4.7 . *Let $G = (V, E)$ be (e, d)-reducible then $\Pi_{cc}^{BW}(G) \leq \frac{e(e+1)}{2} + dn$. In particular, for any DAG G with $\mathsf{indeg}(G) = O(1)$ we have $\Pi_{cc}^{BW}(G) = O\left(\left(\frac{n \log\log n}{\log n}\right)^2\right)$.*

Proof of Theorem 4.7 . Let $S = \{v_1, \ldots, v_e\} \subseteq V$ be given such that $\mathsf{depth}(G - S) \leq d$. For pebbling rounds $i \leq e$ we set $P_i^W = v_i \cup P_i^W$ and $P_i^B = \emptyset$. For pebbling rounds $e < i \leq e + d$ we set $P_i^W = P_{i-1}^W$ and $P_i^B = P_{i-1}^B \cup \{x : \mathsf{parents}(x) \subseteq P_{i-1}^W \cup P_{i-1}^B\}$ so that P_i^B contains every node that can be legally pebbled with a black pebble. Finally, we set $P_{e+d+1} = (\emptyset, \emptyset)$. Clearly, the cost of this pebbling is at most

$$\Pi_{cc}^{BW}(G) \leq dn + \sum_{i=1}^{e} i = dn + \frac{e(e+1)}{2} \ ,$$

since $\left|P_i^W \cup P_i^B\right| = i$ for $i \leq e$, $\left|P_{e+d+1}^W \cup P_{e+d+1}^B\right| = 0$ and we always have $\left|P_i^W \cup P_i^B\right| \leq n$ during any other round i. We now show that the proposed pebbling is legal. We only remove white pebbles during the last round $e+d+1$ so rule (3) is trivially satisfied for rounds $i \leq e+d$. We claim that $P_{e+d}^W \cup P_{e+d}^B = V$. Observe that if this claim is true then rule (5) is satisfied and the last pebbling configuration satisfies rule (3). By definition, the last configuration $P_{e+d+1} = (\emptyset, \emptyset)$ contains no white pebbles so rule (1) is satisfied. Clearly, rounds $i \leq e$ are legal with respect to rules (2) and (4) since we place at most one new white pebble on the

$(a\,|Good_i|\,, (1-a)\,|Good_i| - \epsilon i)$-depth robust. Otherwise, we have a set $S \subseteq V(H_i)$ of size $a\,|Good_i|$ such that $\mathsf{depth}(H_i - S) < (1-a)\,|Good_i| - \epsilon i$ which implies that $\mathsf{depth}(G_n^\epsilon[i] - S) \leq i - |Good_i| + \mathsf{depth}(Good_i - S) < i - a|Good_i| - \epsilon i$ contradicting the depth-robustness of $G_n^\epsilon[i]$.

graph at each point in time. Similarly, during rounds $e+1,\ldots,e+d$ we don't add/remove white pebbles and P_i^B is defined to only include nodes on which a black pebble can be legally pebbled. Thus, rules (2) and (4) are satisfied during all rounds.

It remains to verify that $P_{e+d}^W \cup P_{e+d}^B = V$. To see this we note that at round e we have $\mathsf{depth}(G-(P_e^W \cup P_e^B)) = \mathsf{depth}(G-S) \le d$. We now observe that during each subsequent round the depth is reduced by 1 i.e., for $e < i \le d$ we have $\mathsf{depth}(G-(P_i^W \cup P_i^B)) \le \mathsf{depth}(G-(P_{i-1}^W \cup P_{i-1}^B) - 1$. It follows that $\mathsf{depth}(G-(P_{e+d}^W \cup P_{e+d}^B)) \le 0$, which can only be true if $P_{e+d}^W \cup P_{e+d}^B = V$.

To validate the last claim we simply observe that any DAG G with $\mathsf{indeg}(G) = O(1)$ is (e,d)-reducible with $e = O\left(\frac{n \log\log n}{\log n}\right)$ and $d = O\left(\frac{n}{\log^2 n}\right)$ [AB16].

Reminder of Theorem 4.8. *Let $G = (V=[n], E \supset \{(i,i+1) : i < n\})$ be (e,d)-depth-robust for any $e+d \le (1-\epsilon)n$ then $\Pi_{cc}^{BW}(G) \ge (1/16 - \epsilon/2)\, n^2$. Furthermore, if $G' = ([2n\delta], E')$ is the indegree reduced version of G from Lemma 2.2 then $\Pi_{cc}^{BW}(G') \ge (1/16 - \epsilon/2)\, n^2$. In particular, there is a family of DAGs $\{G_n\}_{n=1}^{\infty}$ with $\mathsf{indeg}(G_n) = O(\log n)$ and $\Pi_{cc}^{BW}(G) = \Omega(n^2)$, and a separate family of DAGs $\{H_n\}_{n=1}^{\infty}$ with $\mathsf{indeg}(H_n) = 2$ and $\Pi_{cc}^{BW}(H_n) = \Omega\left(\frac{n^2}{\log^2 n}\right)$.*

Proof of Theorem 4.8. We first show that $\Pi_{cc}^{BW}(G) \ge (1/16 - \epsilon/2)\, n^2$. Let $P = (P_0, P_1, \ldots, P_t) \in \mathcal{P}G^{BW}$ be given and for simplicity assume that $n/4$ is an integer. Let $B_i = \bigcup_{j \le 4t/n} \left(P_{i+jn/4}^W \cup P_{i+jn/4}^B\right)$ for $i \in [n/4]$. We claim that for each i we have $|B_i| \ge (3/4 - 2\epsilon)n$. If this holds then we have $(3/16 - \epsilon/2)\, n^2 \le \sum_{i \in [n/4]} |B_i| \le \sum_{i \in [t]} \left|P_i^W \cup P_i^B\right|$, and the final claim will follow immediately from Theorem 4.1..

It remains to verify our claim. Consider the interval $[i + jn/4 + 1, i + (j+1)n/4 - 1]$ for some arbitrary j and let $S = \bigcup_{r=i+jn/4+1}^{i+(j+1)n/4-1} P_r^W \setminus P_{i+jn/4}^W$ denote the set of white pebbles placed on $G - B_i$ during this interval. Let $H = \mathsf{ancestors}_{G-B_i}(S)$. Because all white pebbles placed on S were removed by round $i + jn/4$ we note that $H \subseteq \bigcup_{r=i+jn/4+1}^{i+(j+1)n/4-1} \left(P_r^W \cup P_r^B\right)$. Since, H must have been pebbled completely during the interval this means that $\mathsf{depth}(H - S) \le n/4$ since we never place white pebbles on nodes in $V(H-S)$. Thus, H is $(n/4, n/4)$-reducible. On the other hand we note that, by depth-robustness of G, H must be (e,d)-depth-robust for any (e,d) such that $e + d \le |V_H| - \epsilon n$. It follows that $|V_H| \le n(1/2 + \epsilon)$. For any node $x \in V(G - B_i)$ that is pebbled during the interval $[i + jn/4 + 1, i + (j+1)n/4 - 1]$ the length of the longest path to x in $G - B_i$ can be at most $\mathsf{depth}(V_H) + n/4 \le |V_H| + n/4 \le n(1/2+\epsilon) + n/4$. Thus, we have $\mathsf{depth}(G - B_i) \le 3n/4 + \epsilon n$. Since, G is (e,d)-depth-robust for any $e + d \le (1-\epsilon)n$ we must have $|B_i| \ge (1-\epsilon)n - 3n/4 - \epsilon n = n/4 - 2\epsilon n$.

A similar argument shows that $\Pi_{cc}^{BW}(G') \ge (1/16 - \epsilon/2)\, n^2$. Since the argument requires some adaptations we repeat it below for completeness.

Let $P = (P_0, P_1, \ldots, P_t) \in \mathcal{P}{G'}^{BW}$ be given and for simplicity assume that $n/4$ is an integer. Let $B_i' = \bigcup_{j \le 4t/n} \left(P_{i+jn/4}^W \cup P_{i+jn/4}^B\right)$ for $i \in [n/4]$ and let $B_i = \{v \in [n] : [2v\delta + 1, 2(v+1)\delta] \cap B_i' \neq \emptyset\}$ be the corresponding nodes in original DAG G. We claim that for each i we have $|B_i'| \ge (3/4 - 2\epsilon)n$. If this holds then we have

$$(3/16 - \epsilon/2)\, n^2 \le \sum_{i \in [n/4]} |B_i'| \le \sum_{i \in [t]} \left|P_i^W \cup P_i^B\right| \ ,$$

so that $\Pi_{cc}^{BW}(G') \ge (1/16 - \epsilon/2)\, n^2$. The theorem follows immediately from Theorem 4.1 we can take G to be an (e,d)-depth-robust DAG on n nodes with $\mathsf{indeg}(G) = O(\log n)$. G' is now an $n' = 2n\delta = O(n \log n)$ node DAG with $\Pi_{cc}^{BW}(G') = \Omega(n^2) = \Omega(n^2/\log^2 n)$.

To verify that our claim holds consider the interval $[i + jn/4 + 1, i + (j+1)n/4 - 1]$ for some arbitrary j and let $S = \bigcup_{r=i+jn/4+1}^{i+(j+1)n/4-1} P_r^W \setminus P_{i+jn/4}^W$ denote the set of white pebbles placed on $G' - B_i'$ during this interval. Let $H' = \mathsf{ancestors}_{G'-B_i'}(S)$. Because all white pebbles placed on S were removed by round $i + jm$ we note that $H' \subseteq \bigcup_{r=i+jn/4+1}^{i+(j+1)n/4-1} \left(P_r^W \cup P_r^B\right)$. Since, H' must have been pebbled completely during the interval this means that $\mathsf{depth}(H' - S) \le n/4$ since we never place white pebbles on nodes in $V(H' - S)$. Thus, H' is $(n/4, n/4)$-reducible. On the other hand we consider the graph

$H = \mathsf{ancestors}_{G-B_i}\left(\{v : S \cap [2v\delta+1, 2(v+1)\delta] \neq \emptyset\}\right)$. By depth-robustness of G, H must be (e,d)-depth-robust for any (e,d) such that $e+d \leq |V_H| - \epsilon n$. It follows that $|V_H| \leq n(1/2+\epsilon)$. Furthermore, H' is a subgraph of the indegree reduced version of H so $|V_{H'}| \leq \delta |V_H| \leq \delta n(1/2+\epsilon)$.

For any node $x \in V(G' - B_i')$ that is pebbled during the interval $[i + jn/4 + 1, i + (j+1)n/4 - 1]$ the length of the longest path to x in $G' - B_i'$ can be at most $\mathsf{depth}(V_{H'}) + n/4 \leq |V_{H'}| + n/4 \leq \delta n(1/2+\epsilon) + n/4$. Thus, we have $\mathsf{depth}(G' - B_i') \leq \delta n(1/2+\epsilon) + n/4$. Since, G is (e,d)-depth robust for any $e + d \leq (1-\epsilon)n$ it follows from Lemma 2.2 that G' is $(e, d\delta)$-depth-robust for any $e+d \leq (1-\epsilon)n$ [ABP17]. Therefore, we have $|B_i'| \geq (1-\epsilon)n - n/2 - \epsilon n - n/(4\delta) \geq n/4 - 2\epsilon n$.

Theorem A.3 *[PTC76] There is a family of DAGs $\{G_n\}_{n=1}^{\infty}$ with $\mathsf{indeg}(G_n) = 2$ with the property that for some positive constants $c_1, c_2, c_3 > 0$ such that for each $n \geq 1$ the set $S = \{v : \mathsf{parents}(G_n) = \emptyset\}$ of sources has size $|S| \leq c_1 n/\log n$ and for any legal pebbling $P = (P_1, \ldots, P_t) \in \mathcal{P}(G_n)$ there is an interval $[i,j] \subseteq [t]$ during which at least $c_2 n/\log n$ nodes in S are (re)pebbled (formally, $\left|S \cap \bigcup_{k=i}^{j} P_k - P_{i-1}\right| \geq c_2 n/\log n$) and at least $c_3 n/\log n$ pebbles are always on the graph ($\forall k \in [i,j], |P_k| \geq c_3 n/\log n$).*

Reminder of Theorem 3.2. *There is a family of DAGs $\{G_n = (V_n = [n], E_n)\}_{n=1}^{\infty}$ with $\mathsf{indeg}(G_n) = 2$ with the property that for some positive constants $c_1, c_2, c_3 > 0$ such that for each $n \geq 1$ the set $S = \{v \in [n] : \mathsf{parents}(v) = \emptyset\}$ of sources of G_n has size $|S| \leq c_1 n/\log n$ and for any legal pebbling $P = (P_1, \ldots, P_t) \in \mathcal{P}_{G_n}^{\parallel}$ there is an interval $[i,j] \subseteq [t]$ such that (1) $\left|S \cap \bigcup_{k=i}^{j} P_k \setminus P_{i-1}\right| \geq c_2 n/\log n$ i.e., at least $c_2 n/\log n$ nodes in S are (re)pebbled during this interval, and (2) $\forall k \in [i,j], |P_k| \geq c_3 n/\log n$ i.e., at least $c_3 n/\log n$ pebbles are always on the graph.*

Proof of Theorem 3.2. The family of DAGs $\{G_n\}_{n=1}^{\infty}$ is the same as in Theorem A.3. Similarly, let $c_1, c_2, c_3 > 0$ denote the constants from Theorem A.3 and let $S \subseteq V$ be the set of $|S| \leq c_1 n/\log n$ nodes from Theorem A.3.

Let $P = (P_1, \ldots, P_t) \in \mathcal{P}_{G_n}^{\parallel}$ be any pebbling of G_n. We consider the sequential transform $P' = \mathsf{seq}(P) \in \mathcal{P}_{G_n}$ from Definition 2.5. Recall that $P' = (P_1', \ldots, P_{A_t})$ where $A_k = \sum_{i=1}^{k} |P_i \setminus P_{i-1}|$ (and $P_0 \doteq \emptyset$). By Lemma 2.6 P' is a legal sequential pebbling $P' \in \mathcal{P}(G_n)$. Furthermore, we note that $P_{A_i}' = P_i$ for all $i \leq t$ and that $P_i \subset P_{A_i+k}' \subseteq P_i \cup P_{i+1}$ for each $i \leq k$ and $k \leq |P_{i+1} \setminus P_i|$.

Let t_2^* denote the maximum value such that there exists an interval $[t_1^*, t_2^*] \subseteq [A_t]$ such that

$$\left| S \cap \bigcup_{k=t_1^*}^{t_2^*} P_k' - P_{k-1}' \right| \geq c_2 n/\log n \ , \tag{4}$$

and

$$\forall k \in [t_1^*, t_2^*], |P_k'| \geq c_3 n/\log n \ . \tag{5}$$

Observe that by Theorem A.3 t_2^* must exist. Having fixed t_2^* let $t_1^* < t_2^*$ denote the minimum value such that the above properties hold for the interval $[t_1^*, t_2^*]$.

We first claim that $t_2^* = A_j$ for some $j \leq t$. Suppose instead that $t_2^* = A_j + k$ for $0 < k < |P_{j+1} \setminus P_j|$. In this case, we have $P_{t_2^*}' \subsetneq P_{t_2^*+1}'$ which implies that $\left|P_{t_2^*+1}'\right| \geq \left|P_{t_2^*}'\right| \geq c_3' n/\log n$. Furthermore, $\left|S \cap \bigcup_{k=t_1^*}^{t_2^*+1} P_k' - P_{k-1}'\right| \geq \left|S \cap \bigcup_{k=t_1^*}^{t_2^*} P_k' - P_{k-1}'\right| \geq c_2' n/\log n$ so the interval $[t_1^*, t_2^*+1]$ satisfies conditions 4 and 5 above. This contradicts the minimality of t_2^*.

Now suppose that $t_1^* = A_i + k$ for some $0 \leq i \leq t$ and $a_{i+1} > k \geq 0$ and consider the interval $[i + \mathbb{1}_{k>0}, j]$. If $k > 0$ we have the interval $[i+1, j]$. We note that $t_1^* < t_2^*$ and thus $i + 1 \leq j$. Now

$$\left| S \cap \bigcup_{x=i+1}^{j} P_x - P_i \right| = \left| S \cap \bigcup_{x=A_{i+1}}^{A_j} P_x' - P_{A_i}' \right| \geq \left| S \cap \bigcup_{x=t_1^*}^{t_2^*} P_x' - P_{t_1^*-1}' \right| \geq c_2' n/\log n$$

where the second to last inequality follows because $P_{t_1^*-1} = \bigcup_{x=A_i}^{t_1^*-1} P_x$. Furthermore, for each $i < x \le j$ we know that $|P_x| = |P_{A_x}| \ge c_3 n/\log n$ since $t_1^* \le A_x \le t_2^*$. Thus, the interval $[i+1, j] \subseteq [t]$ satisfies both required properties.

If instead $k = 0$ we have the interval $[i, j]$. In this case

$$\left| S \cap \bigcup_{x=i}^{j} P_x - P_{i-1} \right| = \left| S \cap \bigcup_{x=A_i}^{A_j} P'_x - P'_{A_{i-1}} \right| = \left| S \cap \bigcup_{x=t_1^*}^{t_2^*} P'_x - P'_{A_{i-1}} \right| \ge \left| S \cap \bigcup_{x=t_1^*}^{t_2^*} P'_x - P'_{t_1^*-1} \right| \ge c'_2 n/\log n \ ,$$

where the second to last inequality follows since $P'_{t_1^*-1} = P_{A_{i-1}+a_i-1} \supset P_{A_{i-1}}$. Furthermore, for each $i \le x \le j$ we know that $|P_x| = |P_{A_x}| \ge c_3 n/\log n$ since $t_1^* \le A_x \le t_2^*$. $\square$

Claim A.4 *Let G_n^ϵ be an DAG with nodes $V(G_n^\epsilon) = [n]$, indegree $\delta = \mathsf{indeg}(G_n^\epsilon)$ that is (an, bn)-depth robust for all constants $a, b > 0$ such that $a + b \le 1 - \epsilon$, let G be the indegree reduced version of G_n^ϵ from Lemma 2.2 with nodes and $\mathsf{indeg}(G) = 2$ and let $P = (P_1, \ldots, P_t) \in \mathcal{P}_G^{\parallel}$ be a legal pebbling of G such that during some round i the length of the longest unpebbled path in G is at most $\mathsf{depth}(G - P_i) \le c\delta n$ for some constant $1 > c > 0$. Then $\Pi_{ss}(P, n(1-\epsilon-2c)) \ge c\delta n$.*

Proof. Let $k < i$ be the last pebbling step before i during which the length of the longest unpebbled path at time k is at most $\mathsf{depth}(G - P_k) \ge 2cn\delta$. Observe that $k - i \ge \mathsf{depth}(G - P_k) - \mathsf{depth}(G - P_i) \ge cn\delta$ since we can only decrease the depth by at most one in each pebbling round. In particular, $\mathsf{depth}(G - P_k) = 2c\delta$ since $1 + \mathsf{depth}(G - P_k) \le \mathsf{depth}(G - P_{k+1}) < 2cn\delta$. Let $r \in [k, i]$ be given then by construction we have $\mathsf{depth}(G - P_r) \le 2cn\delta$. Let $P'_r = \{v \in V(G_n^\epsilon) : P_r \cap [2\delta(v-1)+1, 2\delta v] \ne \emptyset\}$ be the set of nodes v in G_n^ϵ such that the corresponding path $2\delta(v-1)+1, \ldots, 2\delta v$ contains no pebble at time r. Exploiting the properties of the indegree reduction from Lemma 2.2 we have

$$\mathsf{depth}(G_n^\epsilon - P'_r)\,\delta \le \mathsf{depth}\left(G - \bigcup_{v \in P'_r} [2\delta(v-1)+1, 2\delta v] \right) \le \mathsf{depth}(G - P_r) \le 2cn\delta \ .$$

Now by depth-robustness of G_n^ϵ we have

$$|P'_r| \ge (1-\epsilon)n - \mathsf{depth}(G_n^\epsilon - P'_r) \ge n - \epsilon n - 2cn \ .$$

Thus, $|P_r| \ge |P'_r| \ge n(1-\epsilon-2c)$ for each $r \in [k, i]$. It follows that $\Pi_{ss}(P, n(1-\epsilon-2c)) \ge c\delta n$. $\square$

Reminder of Lemma 2.2. *[ABP17, Lemma 1] (**Indegree-Reduction**) Let $G = (V = [n], E)$ be a (e, d)-depth robust DAG on n nodes and let $\delta = \mathsf{indeg}(G)$. We can efficiently construct a DAG $G' = (V' = [2n\delta], E')$ on $2n\delta$ nodes with $\mathsf{indeg}(G') = 2$ such that for each path $p = (x_1, ..., x_k)$ in G there exists a corresponding path p' of length $\ge k\delta$ in $G'\left[\bigcup_{i=1}^{k}[2(x_i-1)\delta+1, 2x_i\delta]\right]$ such that $2x_i\delta \in p'$ for each $i \in [k]$. In particular, G' is $(e, d\delta)$-depth robust.* The proof of Lemma 2.2 is essentially the same as [ABP17, Lemma 1]. We include it here is the appendix for completeness.

Proof of Lemma 2.2. We identify each node in V' with an element of the set $V \times [2\delta]$ 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 $p_v = (\langle v, 1\rangle, \langle v, 2\rangle, \ldots, \langle v, 2\delta\rangle)$ of length 2δ. 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 $(2\delta)n$.

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 $(\langle u_i, 2\delta\rangle, \langle v, i\rangle)$ to E'. It follows immediately that G' has indegree (at most) 2.

Fix any node set $S' \subset V'$ of size $|S'| \le e$. Then at most e metanodes can share a node with S'. Let $S = \{v \ : \ \exists j \in [2\delta] \ \mathbf{s.t.} \langle v, j\rangle \in S'\}$ denote the set of genesis nodes in G whose metanode shares a node with S' and observe that $|S| \le |S'|$. For each such metanode remove its genesis node in G. Let $p = (v_1, \ldots, v_k)$

be a path in $G - S$. After removing nodes S' from G' there must remain a corresponding path p' in G' running through all the metanodes of p and $|p'| \geq |p|\delta$ since for each v_j p' at minimum contains the nodes $\langle v_j, \delta\rangle, \ldots, \langle v_j, 2\delta\rangle$. In particular, G' must be $(e, d\delta)$-depth robust.

A.1 Proof of Pebbling Reduction

Reminder of Theorem 5.5. *[Pebbling reduction] Let $G_n = (V_n, E_n)$ be a DAG of size $|V_n| = n$. Let $F = (f_{G,n}, \mathcal{N}_{G,n})_{n\in\mathbb{N}}$ be the graph functions for G_n and their naïve oracle algorithms. Then, for any $\lambda \geq 0$, F is an $(\alpha, \beta, \epsilon)$-sustained memory-hard function where*

$$\alpha = \{\alpha_w(q, s, t)\}_{w\in\mathbb{N}} ,$$

$$\beta = \left\{\beta_w(q, s, t) = \frac{\Pi^{\parallel}_{ss}(G, s)(w - \log q)}{1 + \lambda}\right\}_{w\in\mathbb{N}} , \quad \epsilon = \left\{\epsilon_w(q, m) \leq \frac{q}{2^w} + 2^{-\lambda}\right\}_{w\in\mathbb{N}} .$$

Proof of Theorem 5.5. [Sketch] We begin by describing the simulator σ for $\mathbf{r} = (q, s, t)$. Recall that it can make up to $\beta(\mathbf{r})$ calls to $f_n^{(h')}$ (where $h' \leftarrow \mathbb{H}$ is uniform random). Essentially σ runs a copy of algorithm $\mathcal{A}$ on an emulated PROM device parametrized by resource bounds $\mathbf{r}$. For this σ emulates a RO $h \in \mathbb{H}$ to $\mathcal{A}$ as follows. All calls to h are answered consitently with past calls. If the query $\bar{x}$ has not previously been made then σ checks if it has the form $\bar{x} = (x, u, \lambda_1, \lambda_2)$ where all of the following three conditions are met:

1. $u = v_{\mathsf{out}}$ is the sink of G,

2. λ_1 and λ_2 are the labels of the parents of u in G in the (h, x)-labeling of G,

3. $\mathcal{A}$ has already made all other calls to h for the (h, x)-labeling of G in an order respecting the topological sorting of G.

We call a query to h, for which the first two conditions are valid, an *h-final call (for x).* If the third condition also holds then we call the query a *sound* final call. Upon such a fresh final call σ forwards x to $f_n^{(h')}$ to obtain response y. It records $(\bar{x}, y)$ in the function table of h and returns x to $\mathcal{A}$ as the response to its query. If the response from $f_n^{(h')}$ is $\perp$ (because σ has already made $\beta(rv)$ queries) then σ outputs $\perp$ to the distinguisher $\mathcal{D}$ and halts. We must show that $\mathcal{D}$ can not tell an interaction with such an ideal world apart from the real one with greater than probability $\epsilon(\mathbf{l}, \mathbf{r}, n)$.

Next we generalize the pebbling game and notion of sustained space to capture the setting where multiple identical copies of a DAG G are being pebbled. In particular, in our case, when $P = (P_0, P_1, \ldots)$ is a pebbling of m copies of G then we define the *s-block memory* complexity is defined to be $\Pi^{\parallel}_{bm}(P) = \sum \lfloor |P_i|/s \rfloor$. It follows immediatly that G_m consists of m independent copies of G then $\Pi^{\parallel}_{bm}(G_m) \geq m * \Pi^{\parallel}_{ss}(G)$.

The next step in the proof describes a mapping between executions of a pROM algorithm $\mathcal{A}$ and a pebbling of multiple copies of G called the *ex-post-facto* pebbling of the execution. This technique was first used in [DNW05] and has been used in several other pebbling reductions [DKW11, AS15, AT17]. For our case the mapping is identical to that of [AT17] as are the following two key claims. The first states that with high probability (over the choice of coins for $\mathcal{A}$ and choice of the random oracle h) if $\mathcal{A}$ computed m outputs (of distinct inputs) for $f_n^{(h)}$ then ex-post-facto pebbling of that execution will be a legal and complete pebbling of m copies of G. The second claim goes as follows.

Claim A.5 *Fix any input x_{in}. Let σ_i be the i^{th} input state in an execution of $\mathcal{A}^h(x_{\mathsf{in}}; \$)$. Then, for all $\lambda \geq 0$,*

$$\Pr\left[\forall i \ : \ \sum_{x\in X} |P_i^x| \leq \frac{|\sigma_i| + \lambda}{w - \log(q_r)}\right] > 1 - 2^{-\lambda}$$

over the choice of h and \$.

In particular this implies that the size of each individual state in the pROM execution can be upper-bounded by the number of pebbles in the corresponding ex-post-facto pebbling. More generally, the s-block memory complexity of the ex-post-facto pebbling gives us an lower-bound on the s-SMC of the execution. Since the block memory complexity of a graph can be lowerbounded by the sustained space complexity of the graph these results lead to a lowerbound on s-sustained memory complexity of the graph function in terms of the s-sustained space complexity of G.