The Parallel Reversible Pebbling Game: Analyzing the Post-Quantum Security of iMHFs

Jeremiah Blocki, Blake Holman, and Seunghoon Lee

Purdue University, West Lafayette, IN, 47906, USA
{jblocki,holman14,lee2856}@purdue.edu

Abstract. The classical (parallel) black pebbling game is a useful abstraction which allows us to analyze the resources (space, space-time, cumulative space) necessary to evaluate a function f with a static data-dependency graph G. Of particular interest in the field of cryptography are data-independent memory-hard functions $f_{G,H}$ which are defined by a directed acyclic graph (DAG) G and a cryptographic hash function H. The pebbling complexity of the graph G characterizes the amortized cost of evaluating $f_{G,H}$ multiple times as well as the total cost to run a brute-force preimage attack over a fixed domain $\mathcal{X}$, i.e., given $y \in \{0,1\}^*$ find $x \in \mathcal{X}$ such that $f_{G,H}(x) = y$. While a classical attacker will need to evaluate the function $f_{G,H}$ at least $m = |\mathcal{X}|$ times a quantum attacker running Grover's algorithm only requires $\mathcal{O}(\sqrt{m})$ blackbox calls to a quantum circuit $C_{G,H}$ evaluating the function $f_{G,H}$. Thus, to analyze the cost of a quantum attack it is crucial to understand the space-time cost (equivalently width times depth) of the quantum circuit $C_{G,H}$. We first observe that a legal black pebbling strategy for the graph G does not necessarily imply the existence of a quantum circuit with comparable complexity — in contrast to the classical setting where any efficient pebbling strategy for G corresponds to an algorithm with comparable complexity for evaluating $f_{G,H}$. Motivated by this observation we introduce a new parallel reversible pebbling game which captures additional restrictions imposed by the No-Deletion Theorem in Quantum Computing. We apply our new reversible pebbling game to analyze the reversible space-time complexity of several important graphs: Line Graphs, Argon2i-A, Argon2i-B, and DRSample. Specifically, (1) we show that a line graph of size N has reversible space-time complexity at most $\mathcal{O}\left(N^{1+\frac{2}{\sqrt{\log N}}}\right)$. (2) We show that any (e,d)-reducible DAG has reversible space-time complexity at most $\mathcal{O}\left(Ne + dN2^d\right)$. In particular, this implies that the reversible space-time complexity of Argon2i-A and Argon2i-B are at most $\mathcal{O}\left(N^2 \log\log N/\sqrt{\log N}\right)$ and $\mathcal{O}\left(N^2/\sqrt[3]{\log N}\right)$, respectively. (3) We show that the reversible space-time complexity of DRSample is at most $\mathcal{O}\left(N^2 \log\log N/\log N\right)$. We also study the cumulative pebbling cost of reversible pebblings extending a (non-reversible) pebbling attack of Alwen and Blocki on depth-reducible graphs.

Keywords: Parallel Reversible Pebbling · Argon2i · DRSample · Data-Independent Memory-Hard Function

1 Introduction

The (parallel) black pebbling game [PH70, Coo73] is a powerful abstraction which can be used to analyze the resources (space, space-time, amortized space-time) necessary to evaluate any function f_G with a static data-dependency graph G. In the black pebbling game we are given a directed acyclic graph (DAG) $G = (V, E)$ where nodes intuitively represent intermediate data values and edges represent dependencies between these values, e.g., if $z = x \times y$ then we would add directed edges from nodes x and y to node z to indicate that x and y are required to compute z. However, while the parallel black pebbling game is a useful abstraction for classical computation it is not a suitable model for reversible computation as in quantum computation. In this paper, we introduce a parallel reversible pebbling game as an abstraction which can be used to analyze the resources required to build a reversible quantum circuit evaluating our function f_G. We use the parallel

reversible pebbling game to analyze the space-time cost of several important graphs (the line graph, Argon2i-A, Argon2i-B, DRSample) associated with prominent data-independent memory-hard functions (iMHFs) — used in cryptography to design egalitarian proof of work puzzles and to protect low-entropy secrets (e.g., passwords) against brute-force attacks.

Review: Parallel Black Pebbling. The classical parallel black pebbling game begins with no pebbles on the graph ($P_0 = \{\}$), and during each round of the pebbling game, we may only place a new pebble on a node v if all of v's parents were pebbled in the previous round. Intuitively, if the data value X_v corresponding to node v is computed as $X_v := H(X_u, X_{v-1})$ then G would include directed edges (u, v) and $(v-1, v)$ indicating that we cannot compute value X_v (resp. place a pebble on node v) unless X_u and X_{v-1} are already available in memory (resp. we already have pebbles on nodes u and $v-1$). More formally, if $P_i \subseteq V$ denotes the set of pebbled nodes during round i, then we require that $\mathsf{parents}(P_{i+1} \setminus P_i, G) \subseteq P_i$ where $\mathsf{parents}(S, G) = \bigcup_{v \in S}\{u : (u, v) \in E\}$. In the black pebbling game we are given a subset $T \subseteq V$ of target nodes (corresponding to output data values) and the goal of the black pebbling game is to eventually place a pebble on each node in T. A pebbling $P = (P_0, P_1, \ldots, P_t)$ is legal if $P_0 = \{\}$ and $\mathsf{parents}(P_{i+1} \setminus P_i, G) \subseteq P_i$ for each $i < t$. Intuitively, the requirement that $\mathsf{parents}(P_{i+1} \setminus P_i, G) \subseteq P_i$ enforces the natural constraint that we cannot compute a new data value before all dependent data values are available in memory. In the sequential pebbling game, we additionally require that $|P_{i+1} \setminus P_i| \leq 1$ so that only one new pebble can be placed on the graph in each round while the parallel pebbling game has no such restriction. Thus, a legal parallel (resp. sequential) pebbling of a data-dependency graph G naturally corresponds to a parallel (resp. sequential) algorithm to compute f_G and the number of pebbles $|P_i|$ on the graph in each round i corresponds to memory usage during each round of computation.

The sequential black pebbling game has been used to analyze space complexity [HPV77, PTC76] and to examine space-time tradeoffs [Cob66, Coo73, Pau75, PV76, Tom81]. In the field of cryptography, the parallel black pebbling game has been used to analyze the security of data-independent memory-hard functions (iMHFs). An iMHF $f_{G,H}$ is defined using a cryptographic hash function H and a data-dependency graph G [AS15, AB16, ABP17, BZ17]. The output of $f_{G,H}(x)$ is defined to be the label X_N of the final sink node N in G where the label $X_1 = H(X)$ of the first (source) node is obtained by hashing the input and the label of each internal node v is obtained by hashing the labels of all of v's parents, e.g., if $\mathsf{parents}(v, G) = \{u, v-1\}$ then we would set $X_v = H(X_u, x_{v-1})$. In many cryptographic applications (e.g., password hashing), we want to ensure that it is moderately expensive to evaluate $f_{G,H}$ to ensure that a brute-force pre-image attack (given y find some x such that $f_{G,H}(x) = y$) is prohibitively expensive even when the domain $\mathcal{X}$ of inputs is smaller (e.g., low entropy passwords). When modeling the cryptographic hash function H as a random oracle, one can prove that the cost to evaluate $f_{G,H}$ in the parallel random oracle model is exactly captured by the pebbling cost of G [AS15, AT17, ABP18]. Thus, we would like to pick a graph G with high pebbling costs and/or understand the pebbling costs associated with candidate iMHFs. Prior work demonstrated that the amortized space-time complexity of prominent iMHF candidates, including Password Hashing Competition winner Argon2i, was lower than previously hoped [AB16, ABP17, AB17, BZ17]. On the positive side, recent work has shown how to use depth-robust graphs [EGS75] to construct iMHFs with (essentially) optimum amortized space-time complexity [ABP17, ABH17, BHK+19]. However, it is important to note that the classical black pebbling game does not include any rules constraining our ability to remove pebbles. We are allowed to remove pebbles from the graph at any point in time which corresponds to freeing memory

and can be done to reduce the space usage. While the classical pebbling game allows us to discard pebbles at any point in time to free memory, this action is often not possible in a quantum circuit due to the No-Deletion Theorem [KPB00]. In this sense, the black pebbling game cannot be used to model reversible computation as in a quantum circuit and an efficient parallel black pebbling for a graph G does not necessarily imply the existence of a quantum circuit $C_{G,H}$ with comparable cost.

Review: Measuring Pebbling Costs. There are several natural ways to measure the cost of a pebbling. The space cost of a pebbling $P = (P_0, \ldots, P_t)$ measures the maximum number of pebbles on the graph during any round, i.e., $\max_i |P_i|$ and the space complexity of a graph measures the minimum space cost over all legal pebblings of G. Similarly, the space-time cost of a pebbling $P = (P_0, \ldots, P_t)$ measures the product $t \times \max_i |P_i|$ and the cumulative pebbling cost is $\sum_i |P_i|$. Intuitively, space complexity measures the amount of memory (e.g., RAM) required for a computation and space-time cost measures the full cost of the computation by telling how long the memory will be locked up during computation. Cumulative pebbling cost gives the amortized space-time complexity of pebbling multiple copies of the graph G, i.e., when we are evaluating our function f_G on multiple different inputs in parallel [AS15].

(Quantum) Pre-Image Attacks. Understanding the amortized space-time complexity of a graph G is important to estimate the cost of a classical brute-force pre-image attack over a domain $\mathcal{X}$ of size m. In particular, suppose we are given a target output y (e.g., $y = f_{G,H}(x')$ for a secret input $x \in \mathcal{X}$) and we wish to find some input $x' \in \mathcal{X}$ such that $y = f_{G,H}(x')$. Classically, the space-time cost of a black-box pre-image attack would require us to evaluate the function $f_{G,H}$ on $\Omega(m)$ inputs. If the cumulative pebbling cost of G is given by $\sum_i |P_i|$ then the total space-time cost of the pre-image attack would scale proportionally to $m \sum_i |P_i|$, i.e., m times the amortized space-time complexity. Thus, a more efficient black pebbling strategy for G yields a lower-cost pre-image attack.

In the context of quantum computing, Grover's algorithm [Gro96] substantially reduces the cost of a brute-force pre-image attack over a domain $\mathcal{X}$ of size m. In particular, Grover's algorithm only requires $\mathcal{O}(\sqrt{m})$ black-box queries to the function $f_{G,H}$ evaluating the function $f_{G,H}$ and this is optimal — any quantum algorithm using $f_{G,H}$ as a black box must make at least $\Omega(\sqrt{m})$ queries [BBBV97]. If we instantiate $f_{G,H}$ with a quantum circuit of width w and depth d then full Grover circuit would have width $W = \mathcal{O}(w)$ and depth $D = d \times \mathcal{O}(\sqrt{m})$. In particular, the total space-time (equivalently width-depth) cost of the attack would be $wd \times \mathcal{O}(\sqrt{m})$. Thus, to analyze the cost of a quantum pre-image attack it is crucial to understand the space-time (or width-depth) cost of a quantum circuit $C_{G,H}$ computing $f_{G,H}$. Our goal will be to treat H as a black box and use graph pebbling to characterize the space-time cost. A natural first attempt would be to use the classical black pebbling game to analyze the parallel pebbling cost of G as above. If this approach worked we could simply leverage prior (parallel) black pebbling analysis of prominent iMHF candidates [AB16, ABP17, AB17, BZ17] to analyze the cost of a quantum pre-image attack. Unfortunately, this approach breaks down because a legal black pebbling strategy *does not* necessarily correspond to a valid quantum circuit $C_{G,H}$ with comparable cost. Thus, we will require a different pebbling game to analyze the width-depth cost of the quantum circuit $C_{G,H}$.

Notation. We use the notation $[N]$ (resp. $[a, b]$) to denote the set $\{1, \ldots, N\}$ (resp. $\{a, a+1, \ldots, b\}$) for a positive integer N (resp. $a \leq b$). The notation $\xleftarrow{\$}$ denotes a uniformly random sampling, e.g., we say $x \xleftarrow{\$} [N]$ when x is a uniformly sampled integer from 1 to N. For simplicity, we let $\log(\cdot)$ be a log base 2, i.e., $\log x := \log_2 x$.

Let $G = (V, E)$ be a directed acyclic graph (DAG) where we denote N to be the number of nodes in $V = [N]$. Given a node $v \in V$, we define $\mathsf{parents}(v, G)$ to be the *immediate parents* of node v in G, and we extend this definition to a subset of nodes as well; for a set $W \subseteq V$, we define $\mathsf{parents}(W, G) := \bigcup_{w \in W} \{u : (u, w) \in E\}$. We let $\mathsf{ancestors}(v, G)$ be the set of all ancestors of v in G, i.e., $\mathsf{ancestors}(v, G) := \bigcup_{i \geq 1} \mathsf{parents}^i(v, G)$, where $\mathsf{parents}^1(v, G) = \mathsf{parents}(v, G)$ and $\mathsf{parents}^i(v, G) = \mathsf{parents}(\mathsf{parents}^{i-1}(v, G), G)$. Similarly, for a set $W \subseteq V$, we define $\mathsf{ancestors}(W, G) := \bigcup_{i \geq 1} \mathsf{parents}^i(W, G)$, where $\mathsf{parents}^1(W, G) = \mathsf{parents}(W, G)$ and recursively define $\mathsf{parents}^i(W, G) = \mathsf{parents}(\mathsf{parents}^{i-1}(W, G), G)$.

We denote the set of all sink nodes of G with $\mathsf{sinks}(G) := \{v \in V : \nexists (v, u) \in E\}$ – note that $\mathsf{ancestors}(\mathsf{sinks}(G), G) = V$. We define $\mathsf{depth}(v, G)$ to refer to the number of the longest directed path in G ending at node v and we define $\mathsf{depth}(G) = \max_{v \in V} \mathsf{depth}(v, G)$ to refer to the number of nodes in the longest directed path in G. Given a node $v \in V$, we define $\mathsf{indeg}(v) := |\mathsf{parents}(v, G)|$ to denote the number of incoming edges into v, and we also define $\mathsf{indeg}(G) := \max_{v \in V} \mathsf{indeg}(v)$. Given a set $S \subseteq V$ of nodes, we use $G - S$ to refer to the subgraph of G obtained by deleting all the nodes in S and all edges that are incident to S. We also use the notation $S_{\leq k} := S \cap [k]$ denotes the subset of S that only intersects with $[k]$. We say that a DAG $G = (V, E)$ is (e, d)-*depth robust* if for any subset $S \subseteq V$ such that $|S| \leq e$ we have $\mathsf{depth}(G - S) \geq d$. Otherwise, we say that G is (e, d)-*reducible* and call the subset S a *depth-reducing set* (which is of size at most e and yields $\mathsf{depth}(G - S) < d$).

We denote with $\mathcal{P}_{G,T}$ and $\mathcal{P}^{\parallel}_{G,T}$ the set of all legal sequential and parallel *classical* pebblings of G with target set T, respectively. In the case where $T = \mathsf{sinks}(G)$, we simply write $\mathcal{P}_G$ and $\mathcal{P}^{\parallel}_G$, respectively.

1.1 Our Results

We introduce the parallel reversible pebbling game as a tool to analyze the (amortized) space-time cost of a quantum circuit evaluating a function f with a static data-dependency graph G. Prior work [Ben89, Krá01, MSR+19] introduced a sequential reversible pebbling game. As we discuss, there are several key subtleties that arise when extending the sequential reversible pebbling game to the parallel setting. We argue that any parallel reversible pebbling $P = (P_0, \ldots, P_t)$ of the graph G corresponds to a quantum circuit C_P evaluating f with comparable costs, e.g., the depth of the quantum circuit C_P corresponds to the number of pebbling rounds t and the width of the circuit corresponds to the space complexity of the pebbling, i.e., $\max_i |P_i|$. Thus, any reversible pebbling attack will yield a more efficient quantum pre-image attack[1].

As an application, we use the parallel reversible pebbling game to analyze the space-time cost of several important password hashing functions $f_{G,H}$ including PBKDF2, BCRYPT, Argon2i, and DRSample.

Reversible Pebbling Attacks on Line Graphs. We first focus on analyzing the reversible pebbling cost of a line graph L_N with N nodes $\{1, \ldots, N\}$ and edges $(i, i+1)$ for each $1 \leq i < N$. Classically,

[1] While one could use the parallel reversible pebbling game as a heuristic to *lower bound* the cost of a quantum pre-image attack we stress that, at this time, there is no pebbling reduction which provably lower bounds the cost of a quantum pre-image attack on $f_{G,H}$ using reversible pebbling cost of the underlying DAG G. We do have pebbling reductions for classical (non-reversible) pebblings in the parallel random oracle model [AS15], but there are several technical barriers which make it difficult to extend this reduction to the quantum random oracle model.

there is a trivial black pebbling strategy for the line graph with simply walks a single pebble from node 1 to node N over N pebbling rounds, i.e., in each round i we place a new pebble on node i and then delete the pebble on node $i-1$. This pebbling strategy is clearly optimal as the maximum space usage is just 1 and the space-time cost is just $N \times 1 = N$. However, this simple pebbling strategy is no longer legal in the reversible pebbling game and it is a bit tricky just to find a reversible pebbling strategy whose space-time cost is significantly lower than $\mathcal{O}\left(N^2\right)$ — the space-time cost of the naïve pebbling strategy which avoids removing pebbles. In Theorem 1 we show that the (sequential) reversible space-time complexity of a line graph is $\mathcal{O}\left(N^{1+\frac{2}{\sqrt{\log N}}}\right)$. A similar argument seems to be implicitly assumed by Bennett [Ben89] though the argument was never explicitly formalized as a reversible pebbling strategy. The result improves upon a result of Li and Vitányi [LV96] who showed that the space-time complexity is at most $\mathcal{O}\left(N^{\log 3} \log N\right)$[2].

Because the space-time complexity of the line graph $G = L_N$ is so low, it is a poor choice for an iMHF $f_{G,H}$ or for password hashing [BHZ18]. However, the line graph L_N naturally corresponds to widely deployed password hashing algorithms like BCRYPT [PM99] and PBKDF2 [Kal00] which use hash iteration to increase costs where the parameter N controls the number of hash iterations. Thus, to understand the cost of a (quantum) brute-force password cracking attack it is useful to analyze the (reversible) pebbling cost of L_N.

Reversible Pebbling Attack for Depth-Reducible DAGs. In Theorem 2 we give a generic parallel reversible pebbling attack on any (e,d)-reducible DAG G with space-time cost $\mathcal{O}\left(Ne + dN2^d\right)$ which corresponds to a meaningful attack whenever $e = o(N)$ and $d2^d = o(N)$. A DAG G is said to be (e,d)-reducible if there is a subset $S \subseteq V$ of at most e nodes such that any length d path P in G contains at least one node in S. As we show this leads to meaningful reversible pebbling attacks on Argon2i, the winner of the Password Hashing Competition. Specifically, we demonstrate how to construct depth-reducing sets for Argon2i-A (an older version of Argon2i) and Argon2i-B (the current version of Argon2i) with $e = o(N)$ and $d2^d = o(N)$. This leads to reversible pebbling attacks with space-time complexity $\mathcal{O}\left(N^2 \log\log N/\sqrt{\log N}\right)$ and $\mathcal{O}\left(N^2/\sqrt[3]{\log N}\right)$ against Argon2i-A and Argon2i-B, respectively — see Corollary 1.

In the classical pebbling setting, Alwen and Blocki [AB16] previously gave a generic pebbling attack on (e,d)-reducible DAGs with amortized space-time cost $\mathcal{O}\left(Ne + N^2d/e\right)$. However, this pebbling attack is not legal in the reversible setting, and without amortization, the space-time cost is still N^2 — the average number of pebbles on the graph per round is just $e + Nd/e$ but at the peak, the pebbling strategy still requires $\Omega(N)$ pebbles. In our pebbling strategy, the maximum space usage is $\mathcal{O}\left(e + d2^d\right)$.

Reversible Pebbling Attack against DRSample. Finally, we use the parallel reversible pebbling game to analyze DRSample [ABH17] — a proposal to update the edge distribution in Argon2i with a depth-robust graph. With high probability, a randomly sampled DRSample DAG G will not be (e,d)-reducible for parameters e,d as large as $e = \Omega(N/\log N)$ and $d = \Omega(N)$. Thus, the generic reversible pebbling attack on (e,d)-reducible graphs does not seem to apply. We give an alternate pebbling strategy by partitioning the nodes of G into $\lceil N/b \rceil$ consecutive blocks of size b and converting a parallel reversible pebbling of the line graph $L_{\lceil N/b \rceil}$ into a legal reversible pebbling of G. The reversible pebbling strategy will be cost-effective as long as we have an efficient pebbling

[2] The pebbling of Li and Vitányi [LV96] runs in time $\mathcal{O}\left(N^{\log 3}\right)$ while using at most $\mathcal{O}\left(\log N\right)$ pebbles. Our pebbling strategy uses more pebbles to reduce the overall space-time cost by improving the pebbling time.

strategy for $L_{\lceil N/b \rceil}$ and the graph G does not contain too many "long" edges (u, v) with $|v-u| \geq b$ — we show that DRSample does not contain too many long edges when $b = N/\log^2 N$. Combined with our parallel reversible pebbling strategies for the line graph, this leads to an attack on DRSample with space-time cost at most $\mathcal{O}\left(N^2 \log\log N/\log N\right)$ — see Corollary 2.

More generally, in Theorem 3 we give an efficient reversible pebbling algorithm which transforms a legal reversible pebbling $P' = (P'_1, \ldots, P'_{t'})$ of the line graph $L_{\lceil N/b \rceil}$ into a legal reversible pebbling $P = (P_1, \ldots, P_t)$ of a DAG $G = (V, E)$. The reversible pebbling requires $t = \mathcal{O}(bt')$ rounds and space $bs' + (\#skip)$ where $\#skip$ is upper bounded by the number of long edges $(u, v) \in E$ with $|v - u| \geq b$ and $s' = \max_i |P'_i|$ upper bounds the space usage of the pebbling P'. Thus, the total space-time complexity will be $\mathcal{O}\left(b^2 s' t' + N\#skip\right)$ and we will be able to obtain an efficient reversible pebbling attack as long as $b = o(N)$ and $(\#skip) = o(N)$ — we show that this is the case for DRSample.

Cumulative Pebbling Cost and Parallel Reversible Pebbling. Alwen and Blocki [AB16] gave a general parallel black pebbling attack on any (e, d)-reducible graph. This general pebbling attack was used to upper bound the cumulative cost of many prominent iMHFs including Argon2i-A [AB16] and Argon2i-B [AB17]. More generally the attack shows that *any* constant indegree DAG G has cumulative pebbling cost at most $\mathcal{O}\left(N^2 \log\log N/\log N\right)$. We show how the pebbling attack of Alwen and Blocki [AB16] can be extended to the parallel reversible pebbling game[3]. In particular, we can show that the cumulative reversible pebbling costs of an (e, d)-reducible DAG with maximum indegree δ is upper bounded by $\mathcal{O}\left(eN + g\delta N + \frac{N^2 d}{g}\right)$ for any parameter $g \geq d$ matching the non-reversible pebbling attacks of Alwen and Blocki [AB16] — see Theorem 4. More specifically, since any DAG G with constant indegree $\delta = O(1)$ is (e, d)-reducible with $d = N/\log^2 N$ and $e = \mathcal{O}\left(N \log\log N/\log N\right)$ [AB16] we can plug in $g = e$ to obtain a reversible pebbling strategy with cumulative cost at most $\mathcal{O}\left(N^2 \log\log N/\log N\right)$ — see Corollary 4. We can also upper bound the cumulative reversible pebbling costs of Argon2i-A and Argon2i-B as $\mathcal{O}\left(N^{1.75} \log N\right)$ and $\mathcal{O}\left(N^{1.8}\right)$ respectively — see Corollary 3.

1.2 Technical Overview

Defining the Parallel Reversible Pebbling Game. We begin by defining and motivating the parallel reversible pebbling game. We want to ensure that any legal (parallel) reversible pebbling strategy for G corresponds to a quantum circuit $C_{G,H}$ evaluating $f_{G,H}$ that could be used as part of a pre-image attack using Grover's algorithm.

We first consider the parallel quantum random oracle model [BDF+11] where the random oracle is a function $H : \{0,1\}^{\leq 2\lambda} \to \{0,1\}^{\lambda}$. In the parallel quantum random oracle model we are given access to a quantum oracle maps basis states of the form $|x_1, y_1, \ldots, x_k, y_k, z\rangle$ to the new state $|x_1, y_1 \oplus H(x_1), \ldots, x_k, y_k \oplus H(x_k), z\rangle$. Here, $x_1, \ldots, x_k$ denote the queries, $y_1, \ldots, y_k$ denote the output registers and z denotes any auxiliary data. Notice that if $y_i = 0^{\lambda}$ then the i^{th} output register will just be $H(x_i)$ after the query is submitted.

[3] Alwen, Blocki and Pietrzak [ABP17] later provided a recursive version of the pebbling attacks of Alwen and Blocki [AB16] which can further reduces the cumulative pebbling cost of a DAG which is (e_i, d_i)-reducible at a sequence of points (e_i, d_i) with $d_i < d_{i-1}$ and $e_i \geq d_{i-1}$. The recursive pebbling attack yields tighter asymptotic upper bounds for some iMHF candidates [BZ17, ABP17]. We conjecture that these recursive pebbling attacks can also be generalized to the reversible pebbling setting though we leave this as an open problem.

Now consider the function $f(x) = H^N(x)$ where $H^1(x) = H(x)$ and $H^{i+1}(x) = H(H^i(x))$. The data-dependency graph for f is simply the line graph $G = L_N$. In our reversible pebbling game, we want to ensure that each pebbling transition corresponds to a legal state transition in the quantum random oracle model. If $N = 5$, then the pebbling configuration $P_i = \{2, 3, 4\}$ intuitively corresponds to a quantum state containing the labels $X_2 = H^2(x)$, $X_3 = H^3(x)$ and $X_4 = H^4(x)$. From this state, we could use X_4 and an input register and submit the query $|X_4, 0^\lambda\rangle$ to the random oracle to obtain $X_5 = H(X_4)$ from the resulting state $|X_4, H(X_4)\rangle$. Similarly, while we cannot simply delete X_3 we could uncompute this value by using X_3 as an output register and submitting the random oracle query $|X_2, X_3\rangle$ to obtain the new state $|X_2, H(X_2) \oplus X_3\rangle = |X_2, 0^\lambda\rangle$ in which the label X_3 has been removed. However, without the label X_1 there is no way to uncompute X_2 without first recomputing X_1.

The above example suggests that we extend the parallel pebbling game by adding the rule that $\mathsf{parents}(P_i \setminus P_{i+1}, G) \subseteq P_i$, i.e., a pebble can only be deleted if all of its parents were pebbled at the end of the previous pebbling round. While this rule is necessary, it is not yet sufficient to prevent impossible quantum state transitions. In particular, the rule would not rule out the pebbling transition from $P_i = \{1, 2, \ldots, i\}$ to the new configuration $P_{i+1} = \{\}$ where all labels have been removed from memory. This pebbling transition would correspond to a quantum transition from a state in which labels $X_1, \ldots, X_i$ are stored in memory to a new state where all of these labels have been uncomputed after just one (parallel) query to the random oracle. Because quantum computation is reversible this would also imply that we could directly transition from the original state (no labels computed) to a state in which all of the labels $X_1, \ldots, X_i$ are available after just one (parallel) query to the quantum random oracle. However, it is known that computing $X_i = H^i(x)$ requires at least i rounds of computation even in the parallel quantum random oracle model [BLZ21]. Thus, the pebbling transition from $P_i = \{1, 2, \ldots, i\}$ to $P_{i+1} = \{\}$ must be disallowed by our reversible pebbling rules as the corresponding quantum state transition is impossible.

We address this last issue by adding another pebbling rule: if $v \in \mathsf{parents}(P_i \setminus P_{i-1}, G) \cup \mathsf{parents}(P_{i-1} \setminus P_i, G)$, then $v \in P_i$. Intuitively, the rule ensures that if the label X_v appeared in an input register to either compute or uncompute some other data label then we cannot also uncompute X_v in this round, i.e., we must keep a pebble at node v.

We make several observations about the reversible pebbling game. First, any legal reversible pebbling of a DAG G is also a legal (classical) parallel black pebbling of G since we only added additional pebbling restrictions. More formally, if $\mathcal{P}_G^{\parallel}$ (resp. $\mathcal{P}_G$) denotes the set of all legal parallel (resp. sequential) black pebblings of G and $\mathcal{P}_G^{\leftrightarrow,\parallel}$ (resp. $\mathcal{P}_G^{\leftrightarrow}$) denotes the set of all legal parallel (resp. sequential) reversible pebblings of G then we have $\mathcal{P}_G^{\leftrightarrow,\parallel} \subseteq \mathcal{P}_G^{\parallel}$ and $\mathcal{P}_G^{\leftrightarrow} \subseteq \mathcal{P}_G$. Thus, any lower bounds on the classical parallel pebbling cost of G will immediately carry over to the reversible setting. However, upper bounds will not necessarily carry over since classical pebbling attacks may not be legal in the reversible pebbling game. Second, we observe that the following sequential reversible pebbling strategy works for any DAG $G = (V = [N], E)$. In the first N rounds, pebble all nodes in topological order without deleting any pebbles. In the next $N-1$ rounds remove pebbles from all nodes (excluding $\mathsf{sinks}(G)$) in reverse topological order. More formally, assuming that $1, \ldots, N$ is a topological order and that node N is the only sink node we have $P_i = [i]$ for each $i \leq N$ and $P_{N+j} = [N] \setminus [N-j, N-1]$ for each $j \leq N-1$. The pebbling requires N pebbles and finishes in $t = 2N - 1$ rounds so the space-time cost is $2N^2 - N$. We refer to the above sequential strategy as the naïve reversible pebbling for a graph G.

Reversible Pebbling Attack on Line Graphs. We give a reversible pebbling attack on a line graph L_N of size N with the space-time cost $\mathcal{O}\left(N^{1+\frac{2}{\sqrt{\log N}}}\right)$. This can be achieved by generalizing Li and Vitányi's work [LV96]. Li and Vitányi [LV96] gave a reversible pebbling strategy on a line graph of size N with space-time cost $\mathcal{O}\left(N^{\log 3}\log N\right)$ by translating ideas of Bennett [Ben89] into a reversible pebbling argument. Intuitively, if we define $N(k)$ using the recurrence relationship $N(k) = k + \sum_{j=0}^{k-1} N(j)$, solving to $N(k) = 2^k - 1$, then they show that the line graph with $N(k)$ nodes can be pebbled using space $S(k) = S(k-1)+1 = k$ and time $T(k) = 3T(k-1)+1 = \mathcal{O}\left(3^k\right)$ for a total space-time cost of $\mathcal{O}\left(k3^k\right) = \mathcal{O}\left((N(k))^{\log 3}\log N(k)\right)$. Their pebbling strategy works as follows: (1) recursively apply the pebbling strategy to place a pebble on node $N(k-1)$ using space at most $S(k-1)$ and time at most $T(k-1)$, (2) place a pebble on node $v_1 = N(k-1)+1$, (3) recursively apply the strategy (in reverse) to clear any leftover pebbles from nodes 1 to $N(k-1)$ in time $T(k-1)$ and (additional) space at most $S(k-1)$. We are left with $(k-1)+\sum_{j=1}^{k-2} N(j) = N(k-1)$ remaining nodes which will be handled recursively using time $T(k-1)$ and (additional) space $S(k-1)$.

We observe that by increasing the space usage slightly we can decrease the pebbling time to obtain a superior space-time cost. We note that Bennett [Ben89] mentions a similar idea in his paper, but that this idea was not formalized as a reversible pebbling strategy either by Bennett [Ben89] or by Li and Vitányi [LV96]. The key modification is as follows: we redefine $N(k) = ck + \sum_{j=0}^{k-1} cN(j)$ solving to $N(k) = \Theta\left((c+1)^k\right)$. We can now recursively pebble a line graph with $N(k)$ nodes in sequential time $T(k) = (2c+2)T(k-1) + c = \mathcal{O}\left((2c+2)^k\right)$ and space $S(k) = c + S(k-1) = ck$. Intuitively, the recursive pebbling strategy will begin by dropping pebbles on each of the nodes $N(k-1)+1, 2N(k-1)+2, ..., cN(k-1)+c$ using space at most $S(k-1)+c$ and time $2c\cdot T(k-1)$. We are left with $c(k-1)+\sum_{j=0}^{k-2} cN(j) = N(k-1)$ remaining nodes which can then be handled recursively. Setting $c = 2^k$, we have $k = \Theta(\sqrt{\log N(k)})$ yielding an upper bound of $\mathcal{O}\left(N(k)^{1+(2+o(1))\frac{1}{\sqrt{\log N(k)}}}\right)$ on the sequential space-time cost.

We can obtain a minor improvement by exploiting parallelism to save time while increasing space usage slightly. In particular, our parallel strategy uses space $\mathcal{O}\left(c2^k\right)$ and time $\mathcal{O}\left((c+2)^k\right)$ with total space-time cost $\mathcal{O}\left(c(2c+4)^k\right)$. Setting $c+1 = 2^k$ we have a slightly better upper bound $\mathcal{O}\left(N(k)^{1+\frac{2}{\sqrt{\log N(k)}}}\right)$ on the space-time cost. Further details can be found in Appendix A.

Generic Reversible Pebbling Attack on Depth-Reducible Graphs. We give a generic reversible pebbling attack on any (e,d)-reducible DAG $G = (V = [N], E)$ with maximum indegree 2. The space-time cost of our reversible pebbling attack is at most $\mathcal{O}\left(Ne + Nd2^d\right)$. Thus, the attack will be superior to the naïve reversible pebbling strategy as long as $e = o(N)$ and $d2^d = o(N)$. We begin with a depth-reducing set $S \subseteq V$ of size $|S| \leq e$. Our reversible pebbling strategy will never remove pebbles from the set S until all of the sink nodes in G are pebbled and we are ready to remove pebbles from the remaining nodes. On each round $i \leq N$ we will place a new pebble on node $\{i\}$. To ensure that this step is legal, we consider the subgraph formed by all of node i's ancestors in $G - S$. Since $G - S$ does not contain a directed path of length d and each node has at most 2 parents there are at most 2^d ancestors of node i in $G - S$. Once again applying the observation that the depth of $G - S$ is at most d we can start to repebble i's ancestors in round $i - d - 1$ to ensure that i's immediate parents are pebbled by round $i - 1$. After we place a pebble on node i we can remove pebbles from i's ancestors in $G - S$ over the next d rounds. Since we only keep pebbles on

the set S and the ancestors of *up to* $2d$ nodes in $G - S$, the maximum space usage of this reversible pebbling strategy will be $\mathcal{O}\left(e + d2^d\right)$.

We apply the generic attack to Argon2i-A and Argon2i-B. In particular, we apply ideas from the previous work [AB17, BZ17] to show that Argon2i-A (resp. Argon2i-B) graphs are (e, d)-reducible with $e = \mathcal{O}\left(N \log\log N/\sqrt{\log N}\right)$ and $d = \log N/\log\log N$ (resp. $e = \mathcal{O}\left(N/\sqrt[3]{\log N}\right)$ and $d = (\log N)/2$). This leads to reversible pebbling attacks with cost $\mathcal{O}\left(N^2 \log\log N/\sqrt{\log N}\right)$ and $\mathcal{O}\left(N^2/\sqrt[3]{\log N}\right)$) for Argon2i-A and Argon2i-B, respectively. An intriguing open question is whether or not these are the best reversible pebbling attacks for Argon2i-A and Argon2i-B?

Reversible Pebbling Attack on DRSample. We provide a general reversible pebbling attack on any DAG G with the property that G contains few *skip nodes* (defined below). Intuitively, given a DAG $G = (V, E)$ with $|V| = N$ and a parameter $b \geq 1$, we can imagine partitioning the nodes of V into consecutive blocks $B_1 = \{v_1, \ldots, v_b\}, B_2 = \{v_{b+1}, \ldots, v_{2b}\}, \ldots, B_{\lceil N/b \rceil} = \{v_{(\lceil N/b \rceil - 1)b+1}, \ldots, v_N\}$ such that we have $\lceil N/b \rceil$ blocks in total and each block contains exactly b nodes (with the possible exception of the last block if N/b is not an integer). We call a node u in block B_i a *skip node* if G contains a directed edge (u, v) from u to some node $v \in B_j$ with $j > i + 1$ and we call the edge (u, v) a *skip edge*, i.e., the edge (u, v) skips over the block B_{i+1} entirely.

We first observe that if the graph G contained no skip edges then it would be trivial to transform a (parallel) reversible pebbling P' of the line graph $L_{\lceil N/b \rceil} = (V', E')$ with space-time cost $\Pi_{st}^{\leftrightarrow,\parallel}(P')$ into a (parallel) reversible pebbling P of G with space-time cost $\mathcal{O}\left(b^2 \Pi_{st}^{\leftrightarrow,\parallel}(P')\right)$ (see Definition 2 for the definition of $\Pi_{st}^{\leftrightarrow,\parallel}(\cdot)$). In particular, placing a pebbling on node $v' \in V'$ of the line graph corresponds to b rounds in which we pebble all nodes in block $B_{v'}$. Thus, the pebbling time increases by a factor of $\mathcal{O}(b)$, and the total space usage also increases by a factor b. Unfortunately, this strategy may result in an illegal reversible pebbling when G contains skip edges. However, we can modify the above strategy to avoid removing pebbles on skip nodes which intuitively increases our space usage by s — the total number of skip nodes in the graph G. The procedure $P = \mathsf{Trans}(G, P', b)$ is formally described in Algorithm 2 in Appendix D, and an example for the reversible pebbling strategy can be found in in Figure 4 in Appendix B. As long as s is sufficiently small, we obtain an efficient parallel reversible pebbling attack on G. In particular, given a reversible pebbling P' of the line graph $L_{\lceil N/b \rceil} = (V', E')$ with space-time cost $\Pi_{st}^{\leftrightarrow,\parallel}(P')$ we can find a reversible pebbling P of G with space-time cost $\mathcal{O}\left(sN + b^2 \Pi_{st}^{\leftrightarrow,\parallel}(P')\right)$. Combining this observation with our efficient reversible pebbling attacks on the line graph we can see that the space-time costs will be at most $\mathcal{O}\left(sN + b^2 (N/b)^{1+\frac{2}{\sqrt{\log(N/b)}}}\right)$. For graphs like DRSample [ABH17], we can show that (whp) the number of skip nodes is at most $s = \mathcal{O}\left(\frac{N \log\log N}{\log N}\right)$ when we set the block size $b = \mathcal{O}\left(\frac{N}{\log^2 N}\right)$ leading to a reversible pebbling attack with space-time cost $\mathcal{O}\left(\frac{N^2 \log\log N}{\log N}\right)$.

Cumulative Cost for Reversible Pebblings: Depth-Reducing Reversible Pebbling Attacks. Alwen and Blocki [AB16] gave a non-reversible pebbling attack with reduced cumulative pebbling cost for any (e, d)-reducible DAG G. While their pebbling attack is non-reversible, we observe that almost all pebbling rounds respect the constraints of reversible pebbling. We then identify the few non-reversible rounds and how these steps can be patched to respect the additional constraints of reversible pebbling. See details in Section 4.

1.3 Related Work

Related Pebbling Games. Prior work [Ben89, Krá01, MSR+19] introduced a reversible pebbling game to capture restrictions imposed by the Quantum No-Deletion Theorem and analyze space-time tradeoffs in quantum computing. However, the pebbling game considered in these works is sequential and only allows for the addition/removal of one pebble in each round. Thus, the sequential reversible pebbling game is not suitable for analyzing the space-time cost of a quantum circuit evaluating $f_{G,H}$ since the circuit can evaluate H multiple times in parallel. We note that there are several important subtleties that must be considered when extending the game to the parallel setting.

More recently, Kornerup et al. [KSS21] introduced a new (sequential) pebbling game called the *spooky pebble game* to model measurement-based deletion in quantum computation. Intuitively, measurement-based deletion allows for the conversion of some qubits into (cheaper) classical bits which can later be used to restore the quantum state. The spooky pebble game only allows for sequential computation and the cost model ignores classical storage. One disadvantage of instantiating a spooky pebbling attack as part of a quantum pre-image attack is that the final attack requires many intermediate measurements which introduces additional technical challenges, i.e., we need to ensure that *each and every* intermediate measurement does not disturb the state of the nearby qubits or the rest of the quantum computer [Div00]. By contrast, a pebbling attack in our parallel reversible pebbling game naturally corresponds to a quantum circuit which does not require any intermediate measurements and our cost model accounts for the total storage cost (classical + quantum). While Kornerup et al. [KSS21] introduced a spooky pebbling attack on the line graph, we note this spooky pebbling strategy does not yield an efficient reversible pebbling attack in our model as their pebbling attack inherently relies on frequent intermediate measurements to reduce the number of qubits.

Remark 1. One could always try to eliminate the intermediate measurements by applying the "principle of deferred measurement" [NC02]. However, "deferred measurement" increases the space and/or depth of a quantum circuit. For example, if the quantum circuit C acts on s qubits and performs m intermediate measurements then we can obtain an equivalent quantum circuit C' with no intermediate measurements with the caveat that C' operates on $s' = s + \mathsf{poly}(m)$ qubits. The space blowup is especially high if C makes many intermediate measurements, e.g., $s = \mathcal{O}(\log m)$. Fefferman and Remscrim [FR21] gave a space-efficient version of the transform, but their transform yields a large penalty in running time cost, i.e., the transform incurs a multiplicative $\mathsf{poly}(t2^s)$ overhead in the total running time t.

If we apply spooky pebbling in the context of Grover's search then the total number of intermediate measurements m would be exponential, i.e., even if we have a quantum circuit C_f evaluating a function $f : \{0,1\}^k \to \{0,1\}^k$ with just a single intermediate measurement, performing the full Grover's search to find a pre-image of f would involve $m = \mathcal{O}\left(2^{k/2}\right)$ intermediate measurements and applying "deferred measurement" to the full Grover circuit would incur a massive time (or space) penalty. Thus, finding a quantum circuit C_f which has reduced space-time cost and does not require any intermediate measurements would yield a more compelling quantum pre-image attack.

2 Parallel Reversible Pebbling Games

The biggest difference between the classical and reversible pebbling games occurs when removing pebbles from a pebbling configuration. In a classical setting, we can always delete any pebbles in

any point in time when they are no longer needed. On the other hand, in a reversible setting, this is not feasible by quantum no-cloning theorem. Since we can only free a pebble by querying a random oracle at the same input, we can observe that a pebble can be deleted only if we know all of its parents, i.e., all of its parents were previously pebbled. The following definition captures this property:

Definition 1 (Parallel/Sequential Reversible 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) at round i is a subset $P_i \subseteq V$. Let $P = (P_0, \ldots, P_t)$ be a sequence of pebbling configurations. Below are the following properties which define various aspects of reversible pebblings.*

(1) The pebbling should start with no pebbles ($P_0 = \emptyset$) and end with pebbles on all of the target nodes i.e., $T \subseteq P_t$.
(2) A pebble can be added only if all of its parents were pebbled at the end of the previous pebbling round, i.e., $\forall i \in [t] : x \in (P_i \setminus P_{i-1}) \Rightarrow \mathsf{parents}(x, G) \subseteq P_{i-1}$.
(3) (Quantum No-Deletion Property) *A pebble can be deleted only if all of its parents were pebbled at the end of the previous pebbling round, i.e., $\forall i \in [t] : x \in (P_{i-1} \setminus P_i) \Rightarrow \mathsf{parents}(x, G) \subseteq P_{i-1}$.*
(4) (Quantum Reversibility) *If a pebble was required to generate new pebbles (or remove pebbles), then we must keep the corresponding pebble around, i.e., $\forall i \in [t] : x \in \mathsf{parents}(P_i \setminus P_{i-1}, G) \cup \mathsf{parents}(P_{i-1} \setminus P_i, G) \Rightarrow x \in P_i$.*
(5) (Remove Excess Pebbles) *We also consider an optional constraint that $P_t = T$. If a pebbling does not satisfy this optional constraint we call it a relaxed pebbling.*
(6) (Sequential pebbling only) At most one pebble is added or removed in each round, i.e., $\forall i \in [t] : |(P_i \cup P_{i-1}) \setminus (P_i \cap P_{i-1})| \leq 1$.

Now we give pebbling definitions with respect to the above properties.

- *A* legal parallel reversible pebbling *of T is a sequence $P = (P_0, \ldots, P_t)$ of pebbling configurations of G where $P_0 = \varnothing$ and which satisfies conditions (1), (2), (3), (4) and (5) above. If our pebbling additionally satisfies condition (6) then we say that it is a sequential pebbling. Similarly, if our pebbling does not satisfy condition (5) then we call our pebbling strategy a relaxed pebbling.*
- *A* legal reversible pebbling sequence *is a sequence of pebbling configurations $(P_0, \ldots, P_t)$ which satisfies properties (2) and (3) and (4) without requiring $P_0 = \{\}$.*
- *A* legal (non-reversible) pebbling sequence *is a sequence of pebbling configurations $(P_0, \ldots, P_t)$ satisfying condition (2).*

We denote with $\mathcal{P}^{\leftrightarrows}_{G,T}$ and $\mathcal{P}^{\leftrightarrows,\|}_{G,T}$ the set of all legal sequential and parallel reversible pebblings of G with a target set T, respectively. We denote with $\widetilde{\mathcal{P}}^{\leftrightarrows}_{G,T}$ and $\widetilde{\mathcal{P}}^{\leftrightarrows,\|}_{G,T}$ the set of all legal relaxed *sequential and parallel reversible pebblings of G with target set T, respectively. Note that we have $\mathcal{P}^{\leftrightarrows}_{G,T} \subseteq \mathcal{P}^{\leftrightarrows,\|}_{G,T}$ and $\widetilde{\mathcal{P}}^{\leftrightarrows}_{G,T} \subseteq \widetilde{\mathcal{P}}^{\leftrightarrows,\|}_{G,T}$. We will mostly be interested in the case where $T = \mathsf{sinks}(G)$ in which case we simply write $\mathcal{P}^{\leftrightarrows}_{G}$ and $\mathcal{P}^{\leftrightarrows,\|}_{G}$ or $\widetilde{\mathcal{P}}^{\leftrightarrows}_{G}$ and $\widetilde{\mathcal{P}}^{\leftrightarrows,\|}_{G}$, respectively.*

Remark 2. We first note that from any parallel relaxed reversible pebbling of G we can obtain a quantum circuit $C_{G,H}$ which computes $f_{G,H}$. If our pebbling is not relaxed then the circuit $C_{G,H}$ will map the basis state $|x, y, z\rangle$ to the new state $|x, y \oplus f_{G,H}(x), z\rangle$ with no ancilla bits although this property is not necessary for Grover's search. Including the requirement that a reversible pebbling eliminates excess pebbles makes it easier to apply the pebbling attack as a recursive subroutine.

Thus, in this paper, we will focus on finding non-relaxed reversible pebbling attacks. We also note that the space-time cost of a relaxed/non-relaxed reversible pebbling is not fundamentally different. In particular, if $(P_1, \ldots, P_t)$ is a relaxed pebbling where $P_t = T$ contains the final sink node N, then $(P_1, \ldots, P_t, P_{t-1} \cup T, \ldots, P_1 \cup T, T)$ is a legal and complete (non-relaxed) reversible pebbling of G. The running time increases by a multiplicative factor of 2 and the space increases by an additive factor of $|T| \leq |P_t|$ where T is the target set. In particular, the overall space-time costs increase by a multiplicative factor of 4 *at most*. In the remainder of the paper, when we write "legal reversible pebbling" we assume that the pebbling is parallel and non-relaxed by default.

Definition 2 (Reversible Pebbling Complexity). *Given a DAG $G = (V, E)$, we essentially use the same definitions for the reversible pebbling complexity as defined in the previous literature [AS15, ABP17, ABP18]. That is, the standard notion of* time, space, space-time *and* cumulative pebbling complexity (CC) *of a* reversible *pebbling $P = \{P_0, \ldots, P_t\} \in \mathcal{P}_G^{\leftrightarrow,\|}$ are also defined to be:*

- *(time complexity) $\Pi_t^{\leftrightarrow,\|}(P) = t$,*
- *(space complexity) $\Pi_s^{\leftrightarrow,\|}(P) = \max_{i\in[t]} |P_i|$,*
- *(space-time complexity) $\Pi_{st}^{\leftrightarrow,\|}(P) = \Pi_t^{\leftrightarrow,\|}(P) \cdot \Pi_s^{\leftrightarrow,\|}(P)$, and*
- *(cumulative pebbling complexity) $\Pi_{cc}^{\leftrightarrow,\|}(P) = \sum_{i\in[t]} |P_i|$.*

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

$$\Pi_\alpha^{\leftrightarrow,\|}(G, T) = \min_{P\in\mathcal{P}_{G,T}^{\leftrightarrow,\|}} \Pi_\alpha^{\leftrightarrow,\|}(P).$$

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

We define the time, space, space-time *and* cumulative pebbling complexity *of a* sequential *reversible pebbling $P = \{P_0, \ldots, P_t\} \in \mathcal{P}_G^{\leftrightarrow}$ in a similar manner: $\Pi_t^{\leftrightarrow}(P) = t$, $\Pi_s^{\leftrightarrow}(P) = \max_{i\in[t]} |P_i|$, $\Pi_{st}^{\leftrightarrow}(P) = \Pi_t^{\leftrightarrow}(P) \cdot \Pi_s^{\leftrightarrow}(P)$, and $\Pi_{cc}^{\leftrightarrow}(P) = \sum_{i\in[t]} |P_i|$. Similarly, for $\alpha \in \{s, t, st, cc\}$ and a target set $T \subseteq V$, the* sequential *reversible pebbling complexities of G are defined as $\Pi_\alpha^{\leftrightarrow}(G, T) = \min_{P\in\mathcal{P}_{G,T}^{\leftrightarrow}} \Pi_\alpha^{\leftrightarrow}(P)$. When $T = \mathsf{sinks}(G)$ we simplify notation as well and write $\Pi_\alpha^{\leftrightarrow}(G)$.*

When compared to the definition of a *classical* pebbling, we can observe that a reversible pebbling has more restrictions, i.e., it only allows us to have pebbles exactly on the target nodes at the end of the pebbling steps, and it further requires quantum no-deletion property and quantum reversibility. This implies that any legal reversible pebblings are also legal classical pebblings, i.e., $\mathcal{P}_{G,T}^{\|} \subseteq \mathcal{P}_{G,T}^{\leftrightarrow,\|}$ (resp. $\mathcal{P}_{G,T} \subseteq \mathcal{P}_{G,T}^{\leftrightarrow}$). This implies that for any graph G, target set T and cost metric $\alpha \in \{s, t, st, cc\}$, we have $\Pi_\alpha^{\|}(G, T) \leq \Pi_\alpha^{\leftrightarrow,\|}(G, T)$ (resp. $\Pi_\alpha(G, T) \leq \Pi_\alpha^{\leftrightarrow}(G, T)$) for a DAG $G = (V, E)$ and a target set $T \subseteq V$, where $\Pi_\alpha^{\|}(G, T)$ (resp. $\Pi_\alpha(G, T)$) denotes the parallel (resp. sequential) classical pebbling complexities which are defined essentially the same as in Definition 2 with a *classical* pebbling $P = \{P_0, \ldots, P_t\} \in \mathcal{P}_G^{\|}$ (resp. $\mathcal{P}_G$). This means that any lower bound on the classical pebbling complexity of a graph G immediately carries over to the reversible setting and an upper bound (attack) on the reversible pebbling cost immediately carries over to the setting classical pebbling.

In the context of quantum pre-image attacks, parallel space-time costs are arguably the most relevant metric. In particular, the depth of the full Grover circuit scales with the number of queries to our quantum circuit $C_{G,H}$ for $f_{G,H}$ multiplied by the number of pebbling rounds for G. Similarly,

the width of the full Grover circuit will essentially be given by the space usage of our pebbling. Thus, the space-time of Grover's algorithm will scale directly with $\Pi_s^{\leftrightarrow,\|}(P)$. The cumulative pebbling complexity would still be relevant in settings where we are running multiple instances of Grover's algorithm in parallel and can amortize space usage over multiple inputs. In this paper, we primarily focus on analyzing reversible space-time costs, as this would likely be the most relevant metric in practice. However, cumulative pebbling complexity still can be worthwhile to study and we provide some initial results in this direction.

3 Reversible Pebbling Attacks and Applications on iMHFs

3.1 Warmup: Parallel Reversible Pebbling Attack on a Line Graph

We first consider two widely deployed hash functions, PBKDF2 [Kal00] and BCRYPT [PM99], as motivating examples for analyzing a line graph. Basically, they are constructed by hash iterations so they can be modeled as a line graph when simplified. Hence, the pebbling analysis of a line graph tells us about the costs of PBKDF2 and BCRYPT. Although there has been some effort to replace such password-hash functions with memory-hard functions such as Argon2 or SCRYPT [BHZ18], PBKDF2 and BCRYPT are still commonly used by a number of organizations. Thus, it is still important to understand the costs of an offline brute-force attack on passwords protected by functions like PBKDF2 and BCRYPT. In fact, NIST recommends using memory-hard functions for password hashing [GNP+17] but they still allow PBKDF2 and BCRYPT when used with long enough hash iterations. Hence, there is still value to analyze the quantum resistance of these functions. Our reversible pebbling attack on DRSample relies on efficient pebbling strategies for line graphs as a subroutine providing further motivation to understand the reversible pebbling costs of a line graph.

As we illustrated in Section 1.2, we give a (sequential/parallel) reversible pebbling strategy for a line graph L_N using recursion. It can be done by recursively define the sequence of consecutive locations $I(k)$ as $I(k) = I(k-1)' \circ I(k-2)' \circ \ldots \circ I(0)'$ for $k > 0$ and $I(0) = \{\}$, where for $0 \leq j < k$, $I(j)'$ is defined to be a concatenation of c copies of $I(j)$ and i_j (which is an incident node to $I(j)$), i.e., $I(j)' := I(j)^{(1)} \circ i_j^{(1)} \circ I(j)^{(2)} \circ i_j^{(2)} \circ \ldots \circ I(j)^{(c)} \circ i_j^{(c)}$, where $A^{(\ell)}$ denotes the ℓ^{th} copy of A. Intuitively, we can sequentially pebble $I(k)$ by pebbling $I(k-1)', I(k-2)', \ldots, I(0)'$. Here, pebbling $I(j)'$ means that we pebble $I(j)^{(\ell)}$, $i_j^{(\ell)}$, and unpebble $I(j)^{(\ell)}$, and we move on to the next copy to pebble $I(j)^{(\ell+1)}$. We can parallelize this strategy by removing and adding pebbles on the consecutive copies at the same time, which requires more space usage but saves time. Here, we only state the space-time cost of our reversible pebbling strategy on a line graph in Theorem 1. Details of our pebbling strategy can be found in Appendix A.

Theorem 1. *Let L_N be a line graph of size N. Then we have $\Pi_{st}^{\leftrightarrow}(L_N) = \mathcal{O}\left(N^{1+(2+o(1))\frac{1}{\sqrt{\log N}}}\right)$ and $\Pi_{st}^{\leftrightarrow,\|}(L_N) = \mathcal{O}\left(N^{1+\frac{2}{\sqrt{\log N}}}\right)$.*

Proof. The proof directly comes from Lemma 8 in Appendix A. □

3.2 Reversible Pebbling Attacks on (e, d)-reducible DAGs

In this section, we introduce another type of reversible pebbling attack on (e, d)-reducible DAGs with depth-reducing sets with d very small. Recall that a DAG $G = (V, E)$ is (e, d)-reducible if

there exists a subset $S \subseteq V$ with $|S| \leq e$ such that the subgraph $G - S$ does not contain a path of length d. Here, we call such subset S a *depth-reducing set*. In this paper, we only consider DAGs with constant indegree, and especially the current state-of-the-art constructions of iMHFs have indegree 2. Therefore, we will assume that $\mathsf{indeg}(G) = 2$ for the DAGs that we consider.

Since the graph has indegree 2, if we find a depth-reducing set S such that $G - S$ has depth d, then we observe that $|\mathsf{ancestors}(v, G-S)| \leq 2^d$ for any node v in $G-S$. If d is small, i.e., $d \ll \log N$, then $2^d \ll N$ and we can expect that the space-time cost for pebbling such (e, d)-reducible DAG becomes $o(N^2)$. More precisely, we start with giving a regular pebbling strategy (without quantum restrictions) for such DAGs.

Classical Black Pebbling Strategy. We begin by giving a classical pebbling strategy with small space-time complexity. Note that prior pebbling strategies focused exclusively on minimizing cumulative pebbling cost, but the pebbling attacks of Alwen and Blocki [AB16][4] for (e, d)-reducible graphs still have the space-time cost $\Omega(N^2)$.

We first introduce the following helpful notation. For nodes x and y in a DAG $G = (V, E)$, let $\mathsf{LongestPath}_G(x, y)$ denote the number of nodes in the longest path from x to y in G. Then for a node $w \in V$, a depth-reducing set $S \subseteq V$, and a positive integer $i \in \mathbb{Z}_{>0}$, we first define a set $A_{w,S,i}$ which consists of the nodes v where the longest directed path from v to w in $G - S_{\leq w-1}$ has length i, i.e., it contains exactly i nodes.

$$A_{w,S,i} := \left\{ v : \mathsf{LongestPath}_{G-S_{\leq w-1}}(v, w) = i \right\}.$$

It is trivial by definition that for any $v \in V$, $A_{v,S,1} = \{v\}$.

Let $G = (V = [N], E)$ be an (e, d)-reducible DAG. We observe that $\mathsf{depth}(G_{\leq k} - S_{\leq k}) \leq d$ is still true for any $k \leq N$. At round k, we have always ensured that we have pebbles on the set $S_{\leq k}$ and on $\{k\}$ itself. Further, at round k, we can look d steps into the future so that at round $k+d$ we can pebble node $k+d$ without delay. Hence, we start to repebble $\mathsf{ancestors}(k+d, G-S)$ in this round and because $\mathsf{depth}(G_{\leq k} - S_{\leq k}) \leq d$ we are guaranteed to finish within d rounds — just in time to pebble node $k + d$. Taken together, in round k, we have pebbles on $\{k\}$, $S_{\leq k}$, and $\mathsf{ancestors}(k + i, G - S)$ for all $i \leq d$. More precisely, for $v \in V$, let $P_v = S_{\leq v} \cup \left(\bigcup_{j=1}^{d} \bigcup_{i=j}^{d} A_{v-1+j,S,i} \right)$. Since each ancestor graph has size at most 2^d and there are at most d of them, we observe that the total number of pebbles in each round is at most $1 + |S_{\leq k}| + \sum_{i=1}^{d} |\mathsf{ancestors}(k + i, G - S)| \leq 1 + e + d2^d$. Hence, we have that $\Pi^{\parallel}_{st}(G) \leq N(1 + e + d2^d)$.

Reversible Pebbling Strategy. While the above strategy works in the classical setting it will need to be tweaked to obtain a legal reversible pebbling. In particular, after node $k + d$ is pebbled we cannot immediately remove pebbles from all nodes in $\mathsf{ancestors}(k + d, G - S)$ because this would violate our quantum reversibility property. Instead, we can reverse the process and unpebble nodes in $\mathsf{ancestors}(k + d, G - S)$ over the next $G - S$ rounds — with the possible exception of nodes $v \in \mathsf{ancestors}(k + d, G - S)$ which are part of $\mathsf{ancestors}(k + d + j, G - S)$ and are still required for some future node $k + d + j$. Thus, if a DAG G is (e, d)-reducible we can establish the following result.

[4] If G is (e, d)-reducible then Alwen and Blocki [AB16] showed that $\Pi^{\parallel}_{cc}(G) \leq \min_{g \geq d} \left(eN + gN \cdot \mathsf{indeg}(G) + \frac{N^2 d}{g} \right) = o(N^2)$.

Theorem 2. *Let $G = (V = [N], E)$ be an (e, d)-reducible DAG. Then $\Pi_{st}^{\leftrightarrows,\|}(G) = \mathcal{O}\left(Ne + Nd2^d\right)$.*

We will give the proof of Theorem 2 later in the subsection. To prove Theorem 2, we first would need to give a legal reversible pebbling for an (e, d)-reducible DAG G. Lemma 1 provides the desired reversible pebbling for G.

Lemma 1. *Let $G = (V = [N], E)$ be an (e, d)-reducible DAG and let $S \subseteq V$ be a depth-reducing set. Define*

$$B_v := \bigcup_{j=1}^{d+1} \bigcup_{i=j}^{d+1} \left(A_{v+1-j,S,i} \cup A_{v-1+j,S,i}\right),$$

for $v \in V$. Then $P = (P_0, P_1, \ldots, P_{2N})$, where each pebbling configuration is defined by

- *$P_0 = \varnothing$,*
- *for $v \in [N]$, $P_v := S_{\leq v} \cup B_v$, and*
- *for $N < v \leq 2N$, $P_v := P_{2N-v} \cup \{N\}$,*

is a legal parallel reversible pebbling for G.

Before proving Lemma 1, we observe the following key claim. The proof of Claim 1 can be found in Appendix C.

Claim 1. *For $v \in [N]$, $\mathsf{parents}(P_v \setminus P_{v-1}, G) \cup \mathsf{parents}(P_{v-1} \setminus P_v, G) \subseteq P_{v-1} \cap P_v$.*

Proof of Lemma 1: We want to show that it satisfies conditions in Definition 1.

Conditions (1) and (5): $P_{2N} = \{N\}$.

- It is clear that $P_{2N} = P_0 \cup \{N\} = \{N\}$ which is the only target node of the pebbling game.

Condition (2): $\forall v \in [2N] : x \in (P_v \setminus P_{v-1}) \Rightarrow \mathsf{parents}(x, G) \subseteq P_{v-1}$.

- If $v \in [N]$, by Claim 1, we have $\mathsf{parents}(P_v \setminus P_{v-1}) \subseteq P_{v-1} \cap P_v \subseteq P_{v-1}$.
- If $N < v \leq 2N$, we have $P_v \setminus P_{v-1} = (P_{2N-v} \cup \{N\}) \setminus (P_{2N-v+1} \cup \{N\}) = P_{2N-v} \setminus P_{2N-v+1}$. Let $w = 2N - v + 1$, then we have that $w \in [N]$ and $P_v \setminus P_{v-1} = P_{w-1} \setminus P_w$. Now we want to show that $\mathsf{parents}(P_{w-1} \setminus P_w, G) \subseteq P_{v-1} = P_w \cup \{N\}$, which also holds by Claim 1.

Condition (3): $\forall v \in [2N] : x \in (P_{v-1} \setminus P_v) \Rightarrow \mathsf{parents}(x, G) \subseteq P_{v-1}$.

- If $v \in [N]$, by Claim 1, we have $\mathsf{parents}(P_{v-1} \setminus P_v) \subseteq P_{v-1} \cap P_v \subseteq P_{v-1}$.
- If $N < v \leq 2N$, we have $P_{v-1} \setminus P_v = (P_{2N-v+1} \cup \{N\}) \setminus (P_{2N-v} \cup \{N\}) = P_{2N-v+1} \setminus P_{2N-v}$. Then similarly, letting $w = 2N - v + 1$, we have that $w \in [N]$ and $P_{v-1} \setminus P_v = P_w \setminus P_{w-1}$. Now we want to show that $\mathsf{parents}(P_w \setminus P_{w-1}, G) \subseteq P_{v-1} = P_w \cup \{N\}$, which also holds by Claim 1.

Condition (4): $\forall v \in [2N] : x \in \mathsf{parents}(P_v \setminus P_{v-1}, G) \cup \mathsf{parents}(P_{v-1} \setminus P_v, G) \Rightarrow x \in P_v$.

- If $v \in [N]$, this is clear from Claim 1 since $\mathsf{parents}(P_v \setminus P_{v-1}, G) \cup \mathsf{parents}(P_{v-1} \setminus P_v, G) \subseteq P_{v-1} \cap P_v \subseteq P_v$.
- If $N < v \leq 2N$, by similar argument from above, by letting $w = 2N - v + 1$, we have that $w \in [N]$ and $\mathsf{parents}(P_v \setminus P_{v-1}, G) \cup \mathsf{parents}(P_{v-1} \setminus P_v, G) = \mathsf{parents}(P_{w-1} \setminus P_w, G) \cup \mathsf{parents}(P_w \setminus P_{w-1}, G) \subseteq P_{w-1} \cap P_w \subseteq P_{w-1} \subseteq P_{w-1} \cup \{N\} = P_v$.

Taken together, we can conclude that for any $v \in [2N]$, P_v is a legal reversible pebbling configuration for G. □

Now we are ready to prove Theorem 2.

Proof of Theorem 2: Let $P = \{P_0, P_1, \ldots, P_{2N}\}$ as defined in Lemma 1, in which we showed that it is a legal quantum pebbling. Clearly, $\Pi_t^{\leftrightarrow,\|}(P) = 2N$. Further, we observe that $\Pi_s^{\leftrightarrow,\|}(P) \leq \max_{v \in V}\{|S_{\leq v}| + |B_v| + 1\}$. Since we assume that $\mathsf{indeg}(G) = 2$, we have

$$\begin{aligned}
|B_v| &= \left| \bigcup_{j=1}^{d+1} \bigcup_{i=j}^{d+1} (A_{v+1-j,S,i} \cup A_{v-1+j,S,i}) \right| \\
&\leq \sum_{j=1}^{d+1} \sum_{i=j}^{d+1} |A_{v+1-j,S,i}| + |A_{v-1+j,S,i}| \\
&\leq \sum_{j=1}^{d+1} \sum_{i=j}^{d+1} 2^{i+1} = 8d2^d + 2.
\end{aligned}$$

Taken together, $\Pi_{st}^{\leftrightarrow,\|}(P) = \Pi_t^{\leftrightarrow,\|}(P)\Pi_s^{\leftrightarrow,\|}(P) \leq 2N(e + 8d2^d + 3) = \mathcal{O}\left(Ne + Nd2^d\right)$. Hence, we can conclude that $\Pi_{st}^{\leftrightarrow,\|}(G) = \min_{P \in \mathcal{P}_{G,\{N\}}^{\leftrightarrow,\|}} \Pi_{st}^{\leftrightarrow,\|}(P) = \mathcal{O}\left(Ne + Nd2^d\right)$. □

Analysis of Argon2i. There are a number of variants for the Argon2i graphs. We will focus on Argon2i-A [BDK15, BCS16] and Argon2i-B[5] [BDKJ16] here. Recall that Argon2i-A is a graph $G = (V = [N], E)$, where $E = \{(i, i+1) : i \in [N-1]\} \cup \{(r(i), i)\}$, where $r(i)$ is a random value that is picked uniformly at random from $[i-2]$. Argon2i-B has the same structure, except that $r(i)$ is not picked uniformly at random but has a distribution as follows:

$$\Pr[r(i) = j] = \Pr_{x \in [N]}\left[i\left(1 - \frac{x^2}{N^2}\right) \in (j-1, j]\right].$$

Lemma 2. *Let $G_{\mathsf{Arg\text{-}A}} = (V_A = [N], E_A)$ and $G_{\mathsf{Arg\text{-}B}} = (V_B = [N], E_B)$ be randomly sampled graphs according to the Argon2i-A and Argon2i-B edge distributions, respectively. Then with high probability, the following holds:*

(1) $G_{\mathsf{Arg\text{-}A}}$ is (e_1, d_1)-reducible for $e_1 = \frac{N}{d'} + \frac{N \ln \lambda}{\lambda}$ and $d_1 = d'\lambda$, for any $0 < \lambda < N$ and $0 < d' < \frac{N}{\lambda}$.
(2) $G_{\mathsf{Arg\text{-}B}}$ is (e_2, d_2)-reducible for $e_2 = \frac{N}{d'} + \frac{2N}{\sqrt{\lambda}}$ and $d_2 = d'\lambda$, for any $0 < \lambda < N$ and $0 < d' < \frac{N}{\lambda}$.

Alwen and Blocki [AB16, AB17] established similar bounds to Lemma 2, but focused on parameter settings where the depth d is large. By contrast, we will need to pick a depth-reducing set with a smaller depth parameter $d \ll \log N$ to minimize the $d2^d$ cost term in our pebbling attack. The full proof of Lemma 2 can be found in Appendix C. Here, we only give a brief intuition of the proof. To reduce the depth of a graph, we follow the approach of Alwen and Blocki [AB16, AB17] and divide N nodes into λ layers of size N/λ and then reduce the depth of each layer to d' so that the final depth becomes $d = d'\lambda$. To do so, we delete all nodes with parents in the same layer, and

[5] We will follow the naming convention of Alwen and Blocki [AB17] throughout the paper and use Argon2i-A to refer to Argon2i-A v1.1 and Argon2i-B to refer to v1.2+.

then delete one out of d' nodes in each layer. And then we count the number of nodes to be deleted in both steps for each graph.

Applying the result from Lemma 2 to Theorem 2, we have the following space-time cost of reversible pebbling for Argon2i-A and Argon2i-B. Intuitively, we obtain Corollary 1 by setting $\lambda = \sqrt{\log N}$ and $d' = \lambda / \ln \lambda \approx 2\sqrt{\log N}/\log\log N$ (resp. $\lambda = \sqrt[3]{\log^2 N}$ and $d' = \sqrt[3]{\log N}/2$) in Lemma 2 for Argon2i-A (resp. Argon2i-B). The full proof of Corollary 1 can be found in Appendix C.

Corollary 1. *Let $G_{\mathsf{Arg\text{-}A}} = (V_A = [N], E_A)$ and $G_{\mathsf{Arg\text{-}B}} = (V_B = [N], E_B)$ be randomly sampled graphs according to the Argon2i-A and Argon2i-B edge distributions, respectively. Then with high probability, $\Pi_{st}^{\leftrightarrow,\|}(G_{\mathsf{Arg\text{-}A}}) = \mathcal{O}\left(\frac{N^2 \log\log N}{\sqrt{\log N}}\right)$, and $\Pi_{st}^{\leftrightarrow,\|}(G_{\mathsf{Arg\text{-}B}}) = \mathcal{O}\left(\frac{N^2}{\sqrt[3]{\log N}}\right)$.*

Remark 3. Our reversible pebbling attacks on Argon2i-A and Argon2i-B have space-time cost $\Pi_{st}^{\leftrightarrow,\|}(G_{\mathsf{Arg\text{-}A}}) = \mathcal{O}\left(\frac{N^2 \log\log N}{\sqrt{\log N}}\right)$, and $\Pi_{st}^{\leftrightarrow,\|}(G_{\mathsf{Arg\text{-}B}}) = \mathcal{O}\left(\frac{N^2}{\sqrt[3]{\log N}}\right)$ respectively. In the classical setting it was known that $\Pi_{cc}^{\|}(G_{\mathsf{Arg\text{-}A}}) = \tilde{\mathcal{O}}(N^{1.708})$ and $\Pi_{cc}^{\|}(G_{\mathsf{Arg\text{-}B}}) = \tilde{\mathcal{O}}(N^{1.768})$ [ABP17, BZ17]. While these pebbling attacks achieve more impressive reductions in cost, we stress that the attacks are (1) non-quantum (i.e., non-reversible) and (2) the space-time complexity of these classical pebbling attacks is still $\Omega(N^2)$ since there will be a few pebbling rounds with $\Omega(N)$ pebbles on the graph. We remark that since any reversible pebbling is a legal classical pebbling that it immediately follows that $\Pi_{st}^{\|}(G_{\mathsf{Arg\text{-}A}}) = \mathcal{O}\left(\frac{N^2 \log\log N}{\sqrt{\log N}}\right)$, and $\Pi_{st}^{\|}(G_{\mathsf{Arg\text{-}B}}) = \mathcal{O}\left(\frac{N^2}{\sqrt[3]{\log N}}\right)$. The best known classical lower bounds for Argon2i-A and Argon2i-B are $\Pi_{cc}^{\|}(G_{\mathsf{Arg\text{-}A}}) = \Omega(N^{1.66})$ and $\Pi_{cc}^{\|}(G_{\mathsf{Arg\text{-}B}}) = \tilde{\Omega}(N^{1.75})$ which immediately implies that $\Pi_{st}^{\leftrightarrow,\|}(G_{\mathsf{Arg\text{-}A}}) = \Omega(N^{1.66})$, and $\Pi_{st}^{\leftrightarrow,\|}(G_{\mathsf{Arg\text{-}B}}) = \tilde{\Omega}(N^{1.75})$. Thus, there remains a gap between the best upper/lower bounds for $\Pi_{st}^{\leftrightarrow,\|}(G_{\mathsf{Arg\text{-}A}})$ and $\Pi_{st}^{\leftrightarrow,\|}(G_{\mathsf{Arg\text{-}B}})$. Closing or tightening this gap is an interesting open research challenge. Similarly, it would be interesting to figure out if we can find better reversible pebbling strategies to reduce the cumulative cost of Argon2i-A and Argon2i-B, e.g., can one adapt the classical pebbling strategy of the previous work [AB16, ABP17, BZ17] to the reversible setting.

3.3 Reversible Pebbling Attacks using an Induced Line Graph

In this section, we give another general strategy to pebble DAGs by "reducing" the DAG G to a line graph, as shown in Figure 1. Intuitively, given a DAG $G = (V, E)$ with $|V| = N$ and an integer parameter $b \geq 1$, we can partition V into consecutive blocks $B_1, \ldots, B_{\lceil N/b \rceil}$ such that each block contains exactly b nodes, while for the last block we can have less than b nodes if N/b is not an integer.

Notation. Now we consider a reversible pebbling P' of the line graph $L_{\lceil N/b \rceil} = (V' = [\lceil N/b \rceil], E')$. Intuitively, each node in $L_{\lceil N/b \rceil}$ corresponds to each block in G. To transform P' into a pebbling P of G, it will be useful to introduce some notation. Given a node $v' \in V'$ and the pebbling P' of $L_{\lceil N/b \rceil}$, we define $\mathsf{LastDelete}(P', v') := \max\{i : v' \in P'_i\}$ to denote the unique index i such that node $v' \in P'_i$, but $v' \notin P'_j$ for all rounds $j > i$, i.e., the pebble on node v' was removed for the final time in round $i+1$. Similarly, it will be convenient to define $\mathsf{LastAdd}(P') := \max\left\{i : \lceil N/b \rceil \notin P'_{i-1}\right\}$ to be the unique round where a pebble was placed on the last node $v = \lceil N/b \rceil$ for the final time (Note: it is possible that a legal pebbling P' places/removes a pebble on node $v = \lceil N/b \rceil$ several times). We make a couple of basic observations. First, we note that if $u' < v'$ then $\mathsf{LastDelete}(P', u') >$

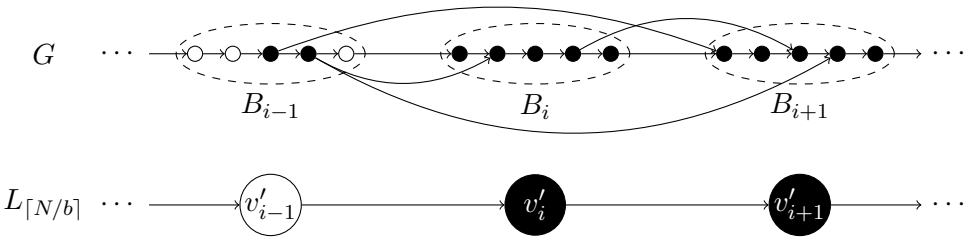

Fig. 1: A line graph $L_{\lceil N/b \rceil}$ induced from a DAG G. Note that each block in an original graph corresponds to a node in the corresponding line graph, e.g., a block B_i in G that consists of five nodes correspond to the node v'_i in $L_{\lceil N/b \rceil}$.

$\mathsf{LastDelete}(P', v')$ since we need node $v'-1$ on the graph to remove a pebble from node v'. Similarly, we note that for any node $v' < \lceil N/b \rceil$ that $\mathsf{LastDelete}(P', v') > \mathsf{LastAdd}(P')$ since we need node $\lceil N/b \rceil - 1$ to be pebbled before we can place a pebble on the final node. Given our graph $G = (V, E)$, a parameter b, and a partition $B_1, \ldots, B_{\lceil N/b \rceil}$ of V into consecutive blocks of size b, we define $\mathsf{Skip}(B_i, G)$, for each i, to be the set of all skip nodes in block B_i, i.e., the set of nodes with an outgoing edge that skips over block B_{i+1}:

$$\mathsf{Skip}(B_i, G) \coloneqq \{v \in B_i : \exists j > i+1 \text{ such that } v \in \mathsf{parents}(B_j, G)\}. \tag{1}$$

We further define $\mathsf{NumSkip}(G, b)$ as the total number of skip nodes in $G = (V, E)$ after partitioning the set of nodes V into consecutive blocks of size b, i.e., $\mathsf{NumSkip}(G, b) \coloneqq \sum_{i=1}^{\lceil N/b \rceil} |\mathsf{Skip}(B_i, G)|$, where B_i's are defined as before.

Pebbling Attempt 1. Our first approach to convert $P' \in \mathcal{P}^{\leftrightarrow,\parallel}_{L_{\lceil N/b \rceil}}$ to a legal reversible pebbling P of G is as follows. Since each node in $L_{\lceil N/b \rceil}$ corresponds to a block (of size at most b) in G, we can transform placing a pebble on a node in $L_{\lceil N/b \rceil}$ to pebbling all nodes in the corresponding block in G in at most b steps. Similarly, we can convert removing a pebble on a node in $L_{\lceil N/b \rceil}$ to removing pebbles from all nodes in the corresponding block in G in at most b steps. It gives us $\mathit{\Pi}^{\leftrightarrow,\parallel}_s(P) \leq b\mathit{\Pi}^{\leftrightarrow,\parallel}_s(P')$ since each node is transformed to a block of size at most b, and $\mathit{\Pi}^{\leftrightarrow,\parallel}_t(P) \leq b\mathit{\Pi}^{\leftrightarrow,\parallel}_t(P')$ since one pebbling/removing step in $L_{\lceil N/b \rceil}$ is transformed to at most b pebbling/removing steps in G.

However, this transformation does *not* yield a legal reversible pebbling of G due to the skip nodes. In particular, given a reversible pebbling configuration $P'_k = \{v'\}$ of $L_{\lceil N/b \rceil}$, it is legal to proceed as $P'_{k+1} = \{v', v'+1\}$. However, when converting it to a reversible pebbling of G, one would need to place pebbles on block $B_{v'+1}$ while only having pebbles on block $B_{v'}$. This could be illegal if there is a node $v \in V$ such that $v \in B_i$ for $i < v'$ and $v \in \mathsf{parents}(B_{v'+1}, G)$, i.e., v is a skip node in B_i, because v must be previously pebbled to place pebbles on block $B_{v'+1}$.

Reversible Pebbling Strategy. To overcome this barrier, when we convert $P' \in \mathcal{P}^{\leftrightarrow,\parallel}_{L_{\lceil N/b \rceil}}$ to a legal reversible pebbling P of G, we define a transformation $P = \mathsf{Trans}(G, P', b)$ which convert placing/removing a pebble on/from a node v' in $L_{\lceil N/b \rceil}$ to placing/removing pebbles on/from all nodes in the corresponding block $B_{v'}$ in G in at most b steps as our first attempt, but when we remove pebbles from $B_{v'}$ in G, we keep skip nodes for the block in the transformation until we delete

pebbles from the block for the last time, i.e., after round $\mathsf{LastDelete}(P', v')$, since these skip nodes will no longer needed to pebble nodes in other blocks in the future.

Furthermore, for the last block (in G), when a pebble is placed on the last node (in $L_{\lceil N/b \rceil}$) for the final time, i.e., in round $\mathsf{LastAdd}(P')$, we indeed want to only pebble the last node (sink node) in the block but not the entire block. Hence, we need additional (at most $b-1$) steps to remove pebbles from all nodes except for the last node in the block.

We can argue the legality of the converted pebbling of G because pebbling steps in each block is legal and keeping skip nodes during the transformation does not affect the legality of pebbling. Intuitively, whenever we pebble a new node v in $L_{\lceil N/b \rceil}$ the node $v-1$ must have been pebbled in the previous round. Thus, in G we will have pebbles on all nodes in the block B_{v-1}. Now for every node $w \in B_v$ and every edge of the form (u, w) we either have (1) $u \in B_{v-1}$, (2) $u \in B_v$ or (3) $u \in B_j$ with $j < v-1$. In the third case, u is a skip node and will already be pebbled allowing us to legally place a pebble on node w. Similarly, in the first case, we are guaranteed that u is already pebbled before we begin pebbling nodes in block B_v since every node in B_{v-1} is pebbled, and in the second case, u will be (re)pebbled before node w. A similar argument shows that all deletions are legal as well. The full proof of Lemma 3 can be found in Appendix C.

Lemma 3. *Let $G = (V = [N], E)$ and $b \in [N]$ be a parameter. If $P' \in \mathcal{P}^{\leftrightarrow,\parallel}_{L_{\lceil N/b \rceil}}$, then $P = \mathsf{Trans}(G, P', b) \in \mathcal{P}^{\leftrightarrow,\parallel}_{G}$.*

The entire procedure $\mathsf{Trans}(G, P', b)$ is formally described in Algorithm 2 in Appendix D, and an example for the reversible pebbling strategy can be found in Figure 4 in Appendix B. Now we observe the following theorem describing the space-time cost of the converted pebbling in terms of the cost of the reduced pebbling of the line graph.

Theorem 3. *Given a DAG $G = (V, E)$ with $|V| = N$ nodes, a reduced line graph $L_{\lceil N/b \rceil} = (V', E')$ with $|V'| = \lceil N/b \rceil$ nodes (where b is a positive integer), and a legal reversible pebbling $P' \in \mathcal{P}^{\leftrightarrow,\parallel}_{L_{\lceil N/b \rceil}}$, there exists a legal reversible pebbling $P = \mathsf{Trans}(G, P', b) \in \mathcal{P}^{\leftrightarrow,\parallel}_{G}$ such that*

$$\Pi^{\leftrightarrow,\parallel}_{st}(P) \leq 2b^2 \Pi^{\leftrightarrow,\parallel}_{st}(P') + 2b\Pi^{\leftrightarrow,\parallel}_{t}(P') \cdot \mathsf{NumSkip}(G, b).$$

Proof. Consider the algorithm $P = \mathsf{Trans}(G, P', b)$ as shown in Algorithm 2 in Appendix D. We argue that the reversible pebbling P is legal in Appendix C and focus here on analyzing the cost of the pebbling P. First, we consider the time cost of P. Notice that in each round P'_j in P' (of the line graph $L_{\lceil N/b \rceil}$), we have two cases: if $j \neq \mathsf{LastAdd}(P', \lceil N/b \rceil)$, we need b rounds to place/remove pebbles in the corresponding blocks in G; otherwise, i.e., $j = \mathsf{LastAdd}(P', \lceil N/b \rceil)$, we need $b + N - (\lceil N/b \rceil - 1)\, b - 1 \leq 2b$ rounds to place/remove pebbles in the corresponding blocks in G. Hence, we have

$$\Pi^{\leftrightarrow,\parallel}_{t}(P) \leq b\left(\Pi^{\leftrightarrow,\parallel}_{t}(P') - 1\right) + 2b = b\left(\Pi^{\leftrightarrow,\parallel}_{t}(P') + 1\right).$$

When it comes to the space cost of the pebbling P, we need space for the pebbling P' multiplied by the block size since each node in P' has a 1-1 correspondence between each block of size b in G. Furthermore, we additionally need space for the skip nodes as they should not be removed to make the pebbling $P = \mathsf{Trans}(G, P', b)$ legal. That is, we have

$$\Pi^{\leftrightarrow,\parallel}_{s}(P) \leq b \cdot \Pi^{\leftrightarrow,\parallel}_{s}(P') + \mathsf{NumSkip}(G, b).$$

Combining these inequalities together, we can conclude that

$$\begin{aligned}\Pi_{st}^{\leftrightarrow,\parallel}(P) &= \Pi_{s}^{\leftrightarrow,\parallel}(P) \cdot \Pi_{t}^{\leftrightarrow,\parallel}(P) \\ &\leq \left(b \cdot \Pi_{s}^{\leftrightarrow,\parallel}(P') + \mathsf{NumSkip}(G,b)\right) \cdot b \left(\Pi_{t}^{\leftrightarrow,\parallel}(P') + 1\right) \\ &= 2b^2 \Pi_{st}^{\leftrightarrow,\parallel}(P') + 2b\Pi_{t}^{\leftrightarrow,\parallel}(P') \cdot \mathsf{NumSkip}(G,b). \end{aligned}$$

□

Analysis on DRSample. DRSample [ABH17] is the first practical construction of an iMHF which modified the edge distribution of Argon2i. Consider a DAG $G = (V = [N], E)$. Intuitively, similar to Argon2i, each node $v \in V \setminus \{1\}$ has at most two parents, i.e., there is a directed edge $(v-1, v) \in E$ and a directed edge from a random predecessor $r(v)$. While Argon2i-A picks $r(v)$ uniformly at random from $[v-2]$, DRSample picks $r(v)$ according to the following random process: (1) We randomly select a bucket index $i \leq \log v$, (2) We randomly sample $r(v)$ from the bucket $B_i(v) = \{u : 2^{i-1} < v - u \leq 2^i\}$. We observe the following lemma which (whp) upper bounds the number of skip nodes when we sample G according to this distribution.

Lemma 4. *Let $G_{\mathsf{DRS}} = (V_{\mathsf{DRS}} = [N], E_{\mathsf{DRS}})$ be a randomly sampled graph according to the DRSample edge distribution. Then with high probability, we have $\mathsf{NumSkip}\left(G_{\mathsf{DRS}}, \left\lceil \frac{N}{\log^2 N} \right\rceil\right) = \mathcal{O}\left(\frac{N \log\log N}{\log N}\right)$.*

The full proof of Lemma 4 can be found in Appendix C. Here, we only give a brief intuition. To count the number of skip nodes, we need to find edges with length longer than b so that the edge skips over a block. There are at most $\log v - \log b$ (out of $\log v$) buckets which potentially could result in a skip node i.e., any edge $(r(v), v)$ with length $v - r(v) \leq b$ cannot produce a new skip node. The probability that the edge $(r(v), v)$ is longer than b is at most $1 - \log b / \log v \leq 1 - \log b / \log N = \log(N/b)/\log N$. Thus, the expected number of skip nodes in DRSample is at most $N \log(N/b)/\log N$ and standard concentration bounds imply that the number of skip nodes will be upper bounded by $\mathcal{O}(N \log(N/b)/\log N)$ with high probability. Setting $b = \lceil N/\log^2 N \rceil$ we can conclude that the expected number of skip nodes in DRSample is at most $\mathcal{O}(N \log\log N/\log N)$ with high probability. Applying the result from Lemma 4 to Theorem 3, we have the following space-time cost of reversible pebbling for DRSample.

Corollary 2. *Let $G_{\mathsf{DRS}} = (V_{\mathsf{DRS}} = [N], E_{\mathsf{DRS}})$ be a randomly sampled graph according to the DRSample edge distribution. Then with high probability, $\Pi_{st}^{\leftrightarrow,\parallel}(G_{\mathsf{DRS}}) = \mathcal{O}\left(\frac{N^2 \log\log N}{\log N}\right)$.*

Proof. Given G_{DRS}, we can consider a reduced line graph $L_{\lceil \log^2 N \rceil} = (V', E')$ with $|V'| = \lceil \log^2 N \rceil$. Then by Theorem 3, we have

$$\Pi_{st}^{\leftrightarrow,\parallel}(G_{\mathsf{DRS}}) \leq 2\left(\frac{N}{\log^2 N}\right)^2 \Pi_{st}^{\leftrightarrow,\parallel}(L_{\lceil \log^2 N \rceil}) + \frac{2N}{\log^2 N} \cdot \Pi_{t}^{\leftrightarrow,\parallel}(L_{\lceil \log^2 N \rceil}) \cdot \mathsf{NumSkip}\left(G_{\mathsf{DRS}}, \left\lceil \frac{N}{\log^2 N} \right\rceil\right).$$

By Theorem 1, we have $\Pi_{st}^{\leftrightarrow,\parallel}(L_{\lceil \log^2 N \rceil}) = \mathcal{O}\left((\log N)^{2\left(1+\frac{2}{\sqrt{\log\log^2 N}}\right)}\right)$ and $\Pi_{t}^{\leftrightarrow,\parallel}(L_{\lceil \log^2 N \rceil}) = \mathcal{O}\left(\log^2 N\right)$. By Lemma 4, we have

$$\Pi_{st}^{\leftrightarrow,\parallel}(G_{\mathsf{DRS}}) \leq 2\left(\frac{N}{\log^2 N}\right)^2 \mathcal{O}\left((\log N)^{2\left(1+\frac{2}{\sqrt{\log\log^2 N}}\right)}\right) + \frac{2N}{\log^2 N} \cdot \mathcal{O}\left(\log^2 N\right) \cdot \mathcal{O}\left(\frac{N \log\log N}{\log N}\right)$$
$$= \mathcal{O}\left(\frac{N^2}{\log^{2(1-\sqrt{2/\log\log N})} N} + \frac{N^2 \log\log N}{\log N}\right) = \mathcal{O}\left(\frac{N^2 \log\log N}{\log N}\right). \qquad \square$$

4 Reversible Pebbling Attacks for Minimizing Cumulative Complexity

In this section, we adapt the depth-reducing pebbling attack GenPeb from Alwen and Blocki [AB16] to a reversible pebbling attack with the same asymptotic CC. The pebbling attack of Alwen and Blocki [AB16] applies to any (e, d)-reducible DAG G with $e = o(N)$ and $d = o(N)$. We first provide an overview of their pebbling strategy before describing how we extend the attack to obtain a reversible pebbling.

Intuitive Overview of [AB16] Attack. Suppose that we are given a DAG $G = (V = [N], E)$ with constant indegree δ along with a depth-reducing set S of size $|S| \leq e$. Intuitively, the pebbling attack of Alwen and Blocki [AB16] can be divided into a series of alternating "light phases" and "balloon phases." It is also helpful to imagine partitioning the nodes $[N]$ into intervals $I_i = [(i-1)g + 1, ig]$ of g consecutive nodes.

- *Light Phases:* During the i^{th} light phase our goal will be to pebble all of the nodes in I_i over the next g consecutive pebbling rounds. The pre-condition for the i^{th} light phase is that we start off with pebbles on all of the nodes $(\mathsf{parents}(I_i) \cup S) \cap [(i-1)g]$ where $\mathsf{parents}(I_i) = \{u : \exists v \in I_i \ \texttt{s.t.}\ (u, v) \in E\}$ denotes the set of parents of nodes in I_i. Similarly, the post-condition for the i^{th} light phase is that we have pebbles on all of the nodes $(\mathsf{parents}(I_i) \cup S) \cap [(i-1)g] \cup I_i$. If $P_j = (\mathsf{parents}(I_i) \cup S) \cap [(i-1)g]$ denotes the initial pebbling configuration at the start of the light phase then we can set $P_{j+x} = P_j \cup [(i-1)g, (i-1)g + x]$ so that P_{j+g} gives us our post-condition. During each light phase we keep *at most* $|(\mathsf{parents}(I_i) \cup S) \cap [(i-1)g] \cup I_i| \leq e + \delta g + g$ pebbles on the graph. Thus, the total cost incurred during each light phase is at most $(e + \delta g + g)g$ and the total cost incurred over all $\frac{N}{g}$ light phases is at most $N(e + \delta g + g)$.
- *Balloon Phases:* The i^{th} balloon phase takes place immediately after the i^{th} light phase with the goal of quickly recovering previously discarded pebbles to satisfy the pre-condition for the next $((i+1)^{st})$ light phase. In particular, the post-condition for the i^{th} balloon phase should match the pre-condition for the $(i+1)^{st}$ light phase. The pre-condition for the i^{th} balloon phase is that our starting configuration contains pebbles on all of the nodes $S \cap [ig]$. During a balloon phase, we are not worried about space so we can recover pebbles on the entire set $[ig]$ within d rounds by exploiting the fact that $G - S$ contains no directed path of length d. Once we have recovered pebbles on the entire set $[ig]$ we can then discard all of the pebbles that are not needed for the next light phase. Thus, the total cost incurred by each individual balloon phase is at most dN and the total cost incurred over all $\frac{N}{g}$ balloon phases is at most $\frac{N^2 d}{g}$.

Formal Description of [AB16] Pebbling Attack. Let $G = ([N], E)$ be an (e, d)-reducible graph and S be a depth-reducing set of size e. The pebbling $P = (P_1, \ldots, P_N)$ from GenPeb lasts N rounds, pebbling each i on round i. The algorithm operates in disjoint and consecutive intervals of $I_c = [(c-1)g + 1, cg]$ where $g \in [d, N]$. At the start of I_c, we perform a "light phase" with the following start and end conditions:

(1) $\mathsf{StartLight}_c \coloneqq P_{(c-1)g+1} = \{(c-1)g + 1\} \cup S_{\leq (c-1)g+1} \cup \mathsf{parents}(I_c)_{\leq (c-1)g+1}$, and
(2) $\mathsf{EndLight}_c \coloneqq P_{cg} \subseteq S_{\leq cg} \cup \{cg\}$.

Intuitively, before we start the light phase, we need to have pebbles on the depth-reducing set, and the parents of the nodes we are about to pebble. By the end of the light phase, all we require for the light phase is that the depth-reducing set and cg is pebbled. As the name suggests, we can define a low-CC pebbling to implement the light phase. For $j \in [g]$ and $k = cg + j$ we define the required pebbles for the j^{th} pebble of the c^{th} light phase as

$$\mathsf{LightReq}_j^c = [(c-1)g + j : k] \cup S_{\leq k} \cup \mathsf{parents}(I_c)_{\leq k}.$$

To pebble the nodes in I_c, we will let $P_{(c-1)g+j} = \mathsf{LightReq}_j^c$. This allows the light phase to only add a pebble to $i \in I_c$ at step i, keeping the overall number of pebbles low. However, this leaves us unprepared for the next light phase (we need to satisfy $\mathsf{StartLight}_{c+1}$ by step $cg + 1$). To fix this, we wait until near the end of the light phase and start a "balloon phase", pebbling as many nodes as possible to quickly pebble a superset of the nodes needed for the next light phase. Since the depth-reducing set $S_{\leq k}$ is pebbled on step k, we can always pebble $G([k])$ by step $k + d$, and then we can simply remove all the pebbles that are unneeded for $\mathsf{StartLight}_{c+1}$. Specifically, the start and end conditions of the balloon phase are

(1) $\mathsf{StartBalloon}_c \coloneqq S_{\leq cg-d+1} \subseteq P_{cg-d+1}$, and
(2) $\mathsf{EndBalloon}_c \coloneqq P_{cg} = [cg]$.

This way at round $cg - d + 1$ we start the balloon phase which pebbles all available nodes for d steps as to eventually satisfy $\mathsf{StartLight}_{c+1}$. Let $R(P_k) = \{v \mid \mathsf{parents}(v) \subseteq P_k\}$ denote the set of nodes that can be pebbled in the next step. Then we can define the balloon requirements per step as $\mathsf{BalloonReq}_1^c = \mathsf{LightReq}_{g-d}^c \cup R(\mathsf{LightReq}_{g-d}^c)$ and for $1 < j \leq d$,

$$\mathsf{BalloonReq}_j^c = \mathsf{BalloonReq}_{j-1}^c \cup R(\mathsf{BalloonReq}_{j-1}^c).$$

Now we can define the low-CC pebbling P such that

$$P_{cg+j} = \begin{cases} \mathsf{LightReq}_j^c & \text{if } j \leq g - d, \text{ and} \\ \mathsf{LightReq}_j^c \cup \mathsf{BalloonReq}_{g-j}^c & \text{otherwise.} \end{cases}$$

It follows that P is a legal pebbling for G [AB16]. We have that $\left|\mathsf{LightReq}_j^c\right| \leq e + g(\delta + 1)$, since each $\mathsf{LightReq}_j^c$ contains at most S, the parents of I_c, and I_c itself. Next we have that each $\left|\mathsf{BalloonReq}_j^c\right| \leq N$, so $\mathit{\Pi}_{cc}^{\|}(P) \leq N\left(\frac{Nd}{g} + e + (\delta + 1)g\right)$. We essentially take the depth-reducing pebbling attack GenPeb from Alwen and Blocki [AB16], and adapt it to be reversible without changing the upper bound asymptotically.

4.1 A Reversible Pebbling Attack

In this section, we define a reversible pebbling extension of GenPeb. We begin with an intuitive overview. We first observe that most pebbling rounds in GenPeb are monotonic, i.e., $P_{i+1} \supset P_i$. Since monotonic transitions do not involve removing pebbles, these transitions remain legal in the reversible pebbling. However, the GenPeb pebbling strategy does include occasionally include a non-monotonic transition at the end of each balloon phase where unnecessary pebbles are simply discarded before the next light phase. Suppose that P_i denotes the pebbling state at the end of the balloon phase and P_{i+1} denotes the pebbling configuration after discarding all of the unnecessary pebbles for the next light phase. The non-monotonic transition from P_i to P_{i+1} will (almost certainly) not be a legal reversible pebbling transition. Our main challenge is to define a legal reversible pebbling sequence which takes us from the pebbling state at the end of each balloon phase to the pebbling state and the beginning of the next light phase. However, while $P_i \not\subseteq P_{i+1}$ we do have $P_{i+1} \subseteq P_i$ since P_{i+1} was obtained by discarding pebbles. Our key idea is to argue that there is a short (i.e., $\leq d$ rounds) monotonic pebbling sequences which takes us P_{i+1} to P_i, i.e., we exploit the fact that any node in P_{i+1} has depth at most d in $G - P_i$ and run a balloon phase. Since this short pebbling sequence is monotonic, it is also reversible. Thus, there is a legal reversible sequence from pebbling state P_i to P_{i+1} in at most d steps.

In the pebbling corresponding to GenPeb, we must remove all unnecessary pebbles after the balloon phase to match the precondition for the following light phase. This is inherently irreversible since they are all removed at once (instead of unpebbling them). For ease of analysis, we start each balloon phase after the corresponding light phase and adjust our notation accordingly. We denote the j^{th} step of the c^{th} light phase as $\mathsf{LightReq}^c_j$ and the balloon phase as $\mathsf{BalloonReq}^c_j$. The light phases themselves remain the same. The first half of each balloon phase remains the same, but $\mathsf{BalloonReq}_{cg-d+j}$ for $j \in [d]$ as defined above. However, we need to "clean up" after each balloon phase in order to meet the precondition for the following light phase, taking care to ensure these pebbling sequences are reversible. In this new balloon phase, we must satisfy the following while maintaining reversibility:

(1) $\mathsf{StartBalloon}_c := S_{\leq cg-d+1} \subseteq \mathsf{BalloonReq}^c_{g-d+1}$,
(2) $\mathsf{MidBalloon}_c := \mathsf{BalloonReq}^c_g = [cg]$, and
(3) $\mathsf{EndBalloon}_c = \mathsf{BalloonReq}^c_{g+1} = \mathsf{LightReq}^{c+1}_1$.

For a sequence of pebbling configurations $P = (P_1, \ldots, P_t)$ let $\mathsf{rev}(P) = (P_t, \ldots, P_1)$. We will use "monotonic" pebbling sequences to generate the reversible pebbling defined above.

Definition 3. *A sequence of pebbling moves $\langle P_1, \ldots, P_t \rangle$ is* monotonic *if $P_1 \subseteq P_2 \subseteq \cdots \subseteq P_t$.*

Immediately, we get that each light phase and balloon phase is monotone. The following result shows that monotonic pebbling sequences are reversible, and the formal proof is left to Appendix C. Intuitively, the additional rules added to the reversible pebbling game only restrict which pebbles we can remove. If a sequence is monotonic then these additional restrictions do not apply.

Lemma 5. *If a legal (non-reversible) pebbling sequence $P = \langle P_1, \ldots, P_t \rangle$ is monotonic, then P is a legal reversible pebbling sequence.*

Now we can use the reverse of the greedy pebbling sequence from $\mathsf{LightReq}_{cg+1}$ to $[cg]$.

Claim 2 (Satisfying $\mathsf{StartLight}_c$ and $\mathsf{EndLight}_c$). *The sequence $(\mathsf{LightReq}^c_1, \ldots, \mathsf{LightReq}^c_g)$ as defined above is a monotonic pebbling sequence.*

Proof. By construction each $\mathsf{LightReq}_i^c \subseteq \mathsf{LightReq}_{i+1}^c$. □

Likewise, the first half of our balloon phase (which is the same as the classical version) is also monotonic, because it simply pebbles all possible nodes each round.

Claim 3 (Satisfying $\mathsf{StartBalloon}_c$ and $\mathsf{MidBalloon}_c$). *The pebbling sequence* $(\mathsf{BalloonReq}_1^c, \ldots, \mathsf{BalloonReq}_d^c)$ *is a legal monotonic pebbling sequence.*

Next, we need to complete the balloon phase. For $j \in [d]$ we let

$$\mathsf{BalloonReq}_{d+j}^c = \mathsf{BalloonReq}_{d-j+1}^c \cup \mathsf{LightReq}_1^{c+1}.$$

Claim 4 (Satisfying $\mathsf{MidBalloon}_c$ and $\mathsf{EndBalloon}_c$). *The pebbling sequence* $(\mathsf{BalloonReq}_{2d}^c, \ldots, \mathsf{BalloonReq}_{d+1}^c)$ *is a legal monotonic pebbling sequence.*

Proof. This follows from the fact that $(\mathsf{BalloonReq}_1^c, \ldots, \mathsf{BalloonReq}_d^c)$ is monotonic. □

Now we can define the first half of our low CC pebbling. Let $\mathsf{LightReq}^c = (\mathsf{LightReq}_1^c, \ldots, \mathsf{LightReq}_g^c)$, $\mathsf{BalloonReq}^c = (\mathsf{BalloonReq}_1^c, \ldots, \mathsf{BalloonReq}_{2d}^c)$, and

$$P_{\text{rev}}^1 = \mathsf{LightReq}^1 + \mathsf{BalloonReq}^1 + \cdots + \mathsf{LightReq}^{\lceil N/g \rceil} + \mathsf{BalloonReq}^{\lceil N/g \rceil},$$

where $+$ denotes sequence concatenation. The sequence P_{rev}^1 is a legal reversible pebbling sequence by the construction of the $\mathsf{LightReq}_j^c$ and $\mathsf{BalloonReq}_j^c$ and by Lemma 6. The proof of Lemma 6 can be found in Appendix C.

Lemma 6. *Let* $\langle P_1, \ldots, P_t \rangle$ *and* $\langle P'_1, \ldots, P'_{t'} \rangle$ *be two legal reversible pebblings for some graph* G *such that* $P_t = P'_{t'}$*. Then for any* $T \subseteq P_t$*,*

$$\langle P_1, \ldots, P_t, P'_{t'-1} \cup T, P'_{t'-2} \cup T, \ldots, P'_1 \cup T \rangle$$

is also a legal reversible pebbling sequence for G*.*

Now we can construct the last part of the pebbling, which simply cleans up by reversing all the prior steps while keeping N pebbled. For a pebbling sequence $Q = (Q_1, \ldots, Q_t)$ and a set K we let $Q(K) = (Q_1 \cup K, \ldots, Q_t \cup K)$. Let

$$P_{\text{rev}}^{2\,\prime} = \mathsf{rev}(\mathsf{BalloonReq}^{\lceil N/g \rceil}) + \mathsf{rev}(\mathsf{LightReq}^{\lceil N/g \rceil - 1}), \ldots, \mathsf{rev}(\mathsf{BalloonReq}^1) + \mathsf{rev}(\mathsf{LightReq}^1) + (\emptyset)$$

and

$$P_{\text{rev}}^2 = P_{\text{rev}}^{2\,\prime}(\{N\}).$$

The final pebbling for G is $P_{\text{rev}} = P_{\text{rev}}^1 + P_{\text{rev}}^2$.

For an arbitrary (e, d)-depth reducible DAG G with depth-reducing set S of size at most e and any $g \in [d, N]$, we let $\mathsf{RGenPeb}(G, e, d, S, g)$ denote the pebbling for G constructed exactly as P_{rev}. The following lemma shows that P_{rev} is a legal reversible pebbling for G. The proof follows from Lemma 6.

Lemma 7. *For any* (e, d)*-depth reducible graph* G *with depth-reducing set* S *of size at most* e*. Then for any* $g \in [d, N]$*,* $P_{rev} = \mathsf{RGenPeb}(G, e, d, S, g)$ *is a legal reversible pebbling for* G*.*

Next we analyze the CC of P_{rev}. This follows similarly to GenPeb, except we need to account for the cost of the extra length of the balloon phases and the cost of having to reverse the pebbling.

Theorem 4. *For any (e,d)-depth reducible graph G on N nodes and any $g \in [d,N]$,*

$$\Pi_{cc}^{\leftrightarrows,\parallel}(G) \leq 2N\left(\frac{2Nd}{g} + e + (\delta+1)g\right) + N + \frac{2Nd}{g}.$$

Proof. We already know that

$$\sum_{c\in[\lceil N/g\rceil]} \sum_{i\in[g]} \left|\mathsf{LightReq}_i^{c-1}\right| \leq N/g(e+(\delta+1)g).$$

Next $\mathsf{BalloonReq}^c$ contains at most $2d$ pebbling steps, so

$$\begin{aligned}\sum_{c\in\lceil N/g\rceil-1} \sum_{i\in[d]} \left|\mathsf{BalloonReq}_i^{c-1}\right| &\leq \sum_{c\in\lceil N/g\rceil-1} 2Nd \\ &\leq N\frac{2Nd}{g}.\end{aligned}$$

Then $\Pi_{cc}^{\leftrightarrows,\parallel}(P_{\text{rev}}^1) \leq N\left(\frac{2Nd}{g} + e + (\delta+1)g\right)$. Next, it's immediate that

$$\Pi_{cc}^{\leftrightarrows,\parallel}(P_{\text{rev}}^2) \leq \Pi_{cc}^{\leftrightarrows,\parallel}(P_{\text{rev}}^1) + \left|P_{\text{rev}}^1\right| \leq N\left(\frac{2Nd}{g} + e(\delta+1)g\right) + N + \frac{2Nd}{g},$$

so

$$\Pi_{cc}^{\leftrightarrows,\parallel}(P_{\text{rev}}) \leq 2N\left(\frac{2Nd}{g} + e + (\delta+1)g\right) + N + \frac{2Nd}{g}. \qquad \square$$

For any iMHF corresponding to a DAG G the reversible cumulative pebbling complexity obtained from our attack is identical to the attack from Alwen and Blocki [AB16]. In particular, for Argon2i-A and Argon2i-B we obtain Corollary 3:

Corollary 3. *Let $G_{\mathsf{Arg\text{-}A}} = (V_A = [N], E_A)$ and $G_{\mathsf{Arg\text{-}B}} = (V_B = [N], E_B)$ be randomly sampled graphs according to the Argon2i-A and Argon2i-B edge distributions, respectively. Then with high probability, we have $\Pi_{cc}^{\leftrightarrows,\parallel}(G_{\mathsf{Arg\text{-}A}}) = \mathcal{O}\left(N^{1.75}\log N\right)$ and $\Pi_{cc}^{\leftrightarrows,\parallel}(G_{\mathsf{Arg\text{-}B}}) = \mathcal{O}\left(N^{1.8}\right)$.*

Proof. Alwen and Blocki [AB16] argued that (whp) a random Argon2i-A DAG $G_{\mathsf{Arg\text{-}A}} = (V_A = [N], E_A)$ is (e,d)-reducible with $d = \sqrt{N}$ and $e = \mathcal{O}\left(N^{0.75}\log N\right)$. The result for Argon2i-A now follows directly from Theorem 4 by setting $g = e$. Alwen and Blocki [AB17] also argued that (whp) a random Argon2i-B DAG $G_{\mathsf{Arg\text{-}B}} = (V_B = [N], E_B)$ is (e,d)-reducible with $d = N^{0.6}$ and $e = \mathcal{O}\left(N^{0.8}\right)$. The result for Argon2i-B now follows directly from Theorem 4 by setting $g = e$. $\square$

Similar to Alwen and Blocki [AB16] we can also obtain a general upper bound for any DAG G with constant indegree.

Corollary 4. *For any DAG $G = (V = [N], E)$ with constant indegree $\delta = \mathcal{O}(1)$ the reversible cumulative pebbling cost at most $\Pi_{cc}^{\leftrightarrows,\parallel}(G) = \mathcal{O}\left(\frac{N^2\log\log N}{\log N}\right)$.*

Proof. Any DAG $G = (V = [N], E)$ with constant indegree $\delta = \mathcal{O}(1)$ is (e,d)-reducible with $d = \frac{N}{\log^2 N}$ and $e = \mathcal{O}\left(\frac{N\log\log N}{\log N}\right)$. The result now follows immediately from Theorem 4 by setting $g = e$. $\square$

5 Conclusion and Open Questions

We introduced the parallel reversible pebbling game and applied it to analyze the reversible space-time complexity of a line graph, Argon2i-A, Argon2i-B, and DRSample. Our motivation is to understand the post-quantum resistance of these MHFs to brute-force pre-image attacks. In particular, we showed that the reversible space-time cost of pebbling a line graph of size N is $\mathcal{O}\left(N^{1+\frac{2}{\sqrt{\log N}}}\right)$ by extending Bennett's reversible pebbling strategy [Ben89]. We also showed that there is a reversible pebbling strategy for an (e, d)-reducible indegree-2 DAG G of size N with the space-time cost $\mathcal{O}\left(Ne + Nd2^d\right)$, which becomes meaningful whenever $e = o(N)$ and $d2^d = o(N)$. We applied this attack to Argon2i-A and Argon2i-B to yield reversible pebbling attacks with space-time cost $\mathcal{O}\left(N^2 \log\log N/\sqrt{\log N}\right)$ and $\mathcal{O}\left(N^2/\sqrt[3]{\log N}\right)$ for Argon2i-A and Argon2i-B, respectively. Finally, we introduced a general reversible pebbling attack on a DAG G of size N by reducing the graph to a line graph $L_{\lceil N/b \rceil}$, and given a legal quantum pebbling P' of the line graph with space-time cost $\Pi_{st}^{\leftrightarrow,\parallel}(P')$, we provided a legal quantum pebbling P of G with space-time cost $\mathcal{O}\left(sN + b^2\Pi_{st}^{\leftrightarrow,\parallel}(P')\right)$, where s denotes the number of skip nodes in G. Tuning the parameter $b = \mathcal{O}\left(N/\log^2 N\right)$ the skip number for DRSample is $\mathcal{O}\left(\frac{N \log\log N}{\log N}\right)$ leading to a reversible pebbling attack with space-time cost $\mathcal{O}\left(N^2 \log\log N/\log N\right)$. We also studied the cumulative pebbling cost of reversible pebblings by extending the depth-reducing attack from Alwen and Blocki [AB16] on depth-reducible graphs.

One open question is to determine if there is a DAG with constant indegree having (parallel) reversible space-time cost $\Omega(N^2)$. Alternatively, is there a generic reversible pebbling attack which rules out this possibility. Blocki et al. [BHK+19] proposed a new iMHF candidate called DRS+BRG (DRSample plus Bit-Reversal Graph) by overlaying a bit-reversal graph [LT82, FLW14] on top of DRSample, which provides the best resistance to known *classical* pebbling attacks. This graph could plausibly have parallel reversible space-time cost $\Omega(N^2)$. In particular, none of the reversible pebbling attacks we proposed perform well against DRS+BRG — there is no small depth-reducing set for DRS+BRG and the extra bit-reversal edges ensure that the number of skip nodes will be large as well.

Another research challenge is to either develop asymptotically stronger reversible pebbling attacks for iMHFs such as Argon2i or establish lower bounds on the parallel reversible space-time complexity. Finally, Alwen et al. [ABP17] defined a recursive (non-reversible) pebbling attack for DAGs that are (e_i, d_i)-depth-reducible for a range of parameters (e_i, d_i) with $e_i > e_{i+1}$ and $d_{i+1} < d_i$. The recursive pebbling attack often leads to improved pebbling attacks with asymptotically lower cumulative pebbling cost (CC). Thus, extending the recursive pebbling attack to the reversible pebbling setting is a natural challenge.

Acknowledgements

Jeremiah Blocki was supported in part by the National Science Foundation under NSF CAREER Award CNS-2047272 and NSF Award CCF-1910659. Seunghoon Lee was supported in part by the Center for Science of Information (NSF CCF-0939370). Blake Holman was supported in part by a Ross Fellowship at Purdue University and by a Ford Foundation Fellowship. We would like to thank anonymous reviewers for helpful feedback which improved this paper.

References

AB16. Joël Alwen and Jeremiah Blocki. Efficiently computing data-independent memory-hard functions. In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part II*, volume 9815 of *LNCS*, pages 241–271. Springer, Heidelberg, August 2016. 2, 3, 5, 6, 9, 14, 16, 17, 21, 22, 25, 26

AB17. Joël Alwen and Jeremiah Blocki. Towards practical attacks on argon2i and balloon hashing. In *Security and Privacy (EuroS&P), 2017 IEEE European Symposium on*, pages 142–157. IEEE, 2017. 2, 3, 6, 9, 16, 25

ABH17. Joël Alwen, Jeremiah Blocki, and Ben Harsha. Practical graphs for optimal side-channel resistant memory-hard functions. In Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu, editors, *ACM CCS 2017*, pages 1001–1017. ACM Press, October / November 2017. 2, 5, 9, 20

ABP17. Joël Alwen, Jeremiah Blocki, and Krzysztof Pietrzak. Depth-robust graphs and their cumulative memory complexity. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *EUROCRYPT 2017, Part III*, volume 10212 of *LNCS*, pages 3–32. Springer, Heidelberg, April / May 2017. 2, 3, 6, 12, 17, 26

ABP18. Joël Alwen, Jeremiah Blocki, and Krzysztof Pietrzak. Sustained space complexity. In Jesper Buus Nielsen and Vincent Rijmen, editors, *EUROCRYPT 2018, Part II*, volume 10821 of *LNCS*, pages 99–130. Springer, Heidelberg, April / May 2018. 2, 12

AS15. Joël Alwen and Vladimir Serbinenko. High parallel complexity graphs and memory-hard functions. In Rocco A. Servedio and Ronitt Rubinfeld, editors, *47th ACM STOC*, pages 595–603. ACM Press, June 2015. 2, 3, 4, 12

AT17. Joël Alwen and Björn Tackmann. Moderately hard functions: Definition, instantiations, and applications. In Yael Kalai and Leonid Reyzin, editors, *TCC 2017, Part I*, volume 10677 of *LNCS*, pages 493–526. Springer, Heidelberg, November 2017. 2

BBBV97. Charles H. Bennett, Ethan Bernstein, Gilles Brassard, and Umesh V. Vazirani. Strengths and weaknesses of quantum computing. *SIAM J. Comput.*, 26(5):1510–1523, 1997. 3

BCS16. Dan Boneh, Henry Corrigan-Gibbs, and Stuart E. Schechter. Balloon hashing: A memory-hard function providing provable protection against sequential attacks. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *ASIACRYPT 2016, Part I*, volume 10031 of *LNCS*, pages 220–248. Springer, Heidelberg, December 2016. 16

BDF+11. Dan Boneh, Özgür Dagdelen, Marc Fischlin, Anja Lehmann, Christian Schaffner, and Mark Zhandry. Random oracles in a quantum world. In Dong Hoon Lee and Xiaoyun Wang, editors, *ASIACRYPT 2011*, volume 7073 of *LNCS*, pages 41–69. Springer, Heidelberg, December 2011. 6

BDK15. Alex Biryukov, Daniel Dinu, and Dmitry Khovratovich. Fast and tradeoff-resilient memory-hard functions for cryptocurrencies and password hashing. Cryptology ePrint Archive, Paper 2015/430, 2015. https://eprint.iacr.org/2015/430. 16

BDKJ16. Alex Biryukov, Daniel Dinu, Dmitry Khovratovich, and Simon Josefsson. The memory-hard argon2 password hash and proof-of-work function. In *Internet-Draft draft-irtf-cfrg-argon2-00, Internet Engineering Task Force*, 2016. 16

Ben89. Charles H. Bennett. Time/space trade-offs for reversible computation. *SIAM J. Comput.*, 18(4):766–776, aug 1989. 4, 5, 8, 10, 26, 29

BHK+19. Jeremiah Blocki, Benjamin Harsha, Siteng Kang, Seunghoon Lee, Lu Xing, and Samson Zhou. Data-independent memory hard functions: New attacks and stronger constructions. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part II*, volume 11693 of *LNCS*, pages 573–607. Springer, Heidelberg, August 2019. 2, 26

BHZ18. Jeremiah Blocki, Benjamin Harsha, and Samson Zhou. On the economics of offline password cracking. In *2018 IEEE Symposium on Security and Privacy*, pages 853–871. IEEE Computer Society Press, May 2018. 5, 13

BLZ21. Jeremiah Blocki, Seunghoon Lee, and Samson Zhou. On the Security of Proofs of Sequential Work in a Post-Quantum World. In Stefano Tessaro, editor, *2nd Conference on Information-Theoretic Cryptography (ITC 2021)*, volume 199 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 22:1–22:27, Dagstuhl, Germany, 2021. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. 7

BZ17. Jeremiah Blocki and Samson Zhou. On the depth-robustness and cumulative pebbling cost of Argon2i. In Yael Kalai and Leonid Reyzin, editors, *TCC 2017, Part I*, volume 10677 of *LNCS*, pages 445–465. Springer, Heidelberg, November 2017. 2, 3, 6, 9, 17

Cob66. Alan Cobham. The recognition problem for the set of perfect squares. In *7th Annual Symposium on Switching and Automata Theory (swat 1966)*, pages 78–87, 1966. 2

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, page 29–33, New York, NY, USA, 1973. Association for Computing Machinery. 1, 2

Div00. David P. Divincenzo. The physical implementation of quantum computation. *Fortschr. Phys*, 48:2000, 2000. 10

EGS75. P. Erdös, R.L. Graham, and E. Szemerédi. On sparse graphs with dense long paths. *Computers & Mathematics with Applications*, 1(3):365 – 369, 1975. 2

FLW14. Christian Forler, Stefan Lucks, and Jakob Wenzel. Memory-demanding password scrambling. In Palash Sarkar and Tetsu Iwata, editors, *ASIACRYPT 2014, Part II*, volume 8874 of *LNCS*, pages 289–305. Springer, Heidelberg, December 2014. 26

FR21. Bill Fefferman and Zachary Remscrim. Eliminating intermediate measurements in space-bounded quantum computation. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2021, page 1343–1356, New York, NY, USA, 2021. Association for Computing Machinery. 10

GNP+17. Paul Grassi, Elaine Newton, Ray Perlner, Andrew Regenscheid, William Burr, Justin Richer, Naomi Lefkovitz, Jamie Danker, Yee-Yin Choong, Kristen Greene, and Mary Theofanos. Digital identity guidelines: Authentication and lifecycle management, 2017-06-22 2017. 13

Gro96. Lov K. Grover. A fast quantum mechanical algorithm for database search. In *28th ACM STOC*, pages 212–219. ACM Press, May 1996. 3

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

Kal00. Burt Kaliski. PKCS #5: Password-Based Cryptography Specification Version 2.0. RFC 2898, RSA Laboratories, September 2000. 5, 13

KPB00. A. Kumar Pati and S. Braunstein. Impossibility of deleting an unknown quantum state. *Nature*, 404:164–165, 2000. 3

Krá01. Richard Král'ovič. Time and space complexity of reversible pebbling. In Leszek Pacholski and Peter Ružička, editors, *SOFSEM 2001: Theory and Practice of Informatics*, pages 292–303, Berlin, Heidelberg, 2001. Springer Berlin Heidelberg. 4, 10

KSS21. Niels Kornerup, Jonathan Sadun, and David Soloveichik. The spooky pebble game, 2021. 10

LT82. Thomas Lengauer and Robert E. Tarjan. Asymptotically tight bounds on time-space trade-offs in a pebble game. *J. ACM*, 29(4):1087–1130, October 1982. 26

LV96. Ming Li and Paul Vitányi. Reversibility and adiabatic computation: Trading time and space for energy. *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, 452(1947):769–789, Apr 1996. 5, 8, 29

MSR+19. Giulia Meuli, Mathias Soeken, Martin Roetteler, Nikolaj Bjorner, and Giovanni De Micheli. Reversible pebbling game for quantum memory management. In *2019 Design, Automation Test in Europe Conference Exhibition (DATE)*, pages 288–291, 2019. 4, 10

NC02. Michael A Nielsen and Isaac Chuang. Quantum computation and quantum information, 2002. 10

Pau75. Wolfgang J. Paul. A 2.5 n-lower bound on the combinational complexity of boolean functions. In *Proceedings of the Seventh Annual ACM Symposium on Theory of Computing*, STOC '75, page 27–36, New York, NY, USA, 1975. Association for Computing Machinery. 2

PH70. Michael S. Paterson and Carl E. Hewitt. *Comparative Schematology*, page 119–127. Association for Computing Machinery, New York, NY, USA, 1970. 1

PM99. Niels Provos and David Mazières. A future-adaptive password scheme. In *Proceedings of the Annual Conference on USENIX Annual Technical Conference*, ATEC '99, page 32, USA, 1999. USENIX Association. 5, 13

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, page 149–160, New York, NY, USA, 1976. Association for Computing Machinery. 2

PV76. Nicholas Pippenger and Leslie G. Valiant. Shifting graphs and their applications. *J. ACM*, 23(3):423–432, July 1976. 2

Tom81. Martin Tompa. Corrigendum: Time-space tradeoffs for computing functions, using connectivity properties of their circuits. *J. Comput. Syst. Sci.*, 23(1):106, 1981. 2

A Reversible Pebbling Strategies on a Line Graph

In this section, we review the (sequential) reversible pebbling strategy for a line graph L_N from Li and Vitányi [LV96] which translated Bennett's reversible simulation [Ben89] into a (sequential) reversible pebbling game, and we give a reversible pebbling strategy with a better space-time cost. We remark that a similar argument seems to be implicitly assumed in Bennett [Ben89], though no explicit description of the reversible pebbling is provided. Hence, we include this result for completeness.

The strategy from Li and Vitányi [LV96] is as follows: let $I(k) = I(k-1) \circ i_{k-1} \circ I(k-2) \circ i_{k-2} \circ \ldots \circ I_1 \circ i_1 \circ I_0 \circ i_0$, where for $j = 0, 1, \ldots, k$, $I(j)$ denotes the sequence of consecutive locations in L_N, $I(0) = \{\}$, and i_j denotes the node incident to $I(j)$. Let $N(k)$ be the size of $I(k)$. Then we have $N(k) = \sum_{i=1}^{k-1}(N(i)+1)$ with $N(0) = 0$, which implies $N(k) = 2^k - 1$. The reversible pebbling works as we pebble the block $I(j)$, pebble i_j, and unpebble $I(j)$. If P denotes such reversible pebbling, then Li and Vitányi [LV96] showed that $P \in \mathcal{P}^{\leftrightarrows}_{I(k)}$ and $\Pi^{\leftrightarrows}_s(P) = \mathcal{O}(\log N(k))$ and $\Pi^{\leftrightarrows}_t(P) = \mathcal{O}\left(N(k)^{\log 3}\right)$, since if we denote $S(k)$ (resp. $T(k)$) the space (resp. time) cost to reversibly pebble $I(k)$ then it satisfies the recurrence relation $S(k) = \max_i\{i + S(k-i)\} = S(k-1)+1$ (resp. $T(k) = 2T(k-1) + 1 + 2T(k-2) + 1 + \ldots + 2T(1) + 1 = 3T(k-1) + 1$). Taken together, the reversible space-time cost for this pebbling strategy is $\Pi^{\leftrightarrows}_{st}(P) = \mathcal{O}\left(N(k)^{\log 3} \log N(k)\right)$, which implies that $\Pi^{\leftrightarrows}_{st}(L_N) = \mathcal{O}\left(N^{\log 3} \log N\right)$.

A Reversible Pebbling Strategy with a Better ST Cost. We extend this approach and first recursively define the sequence of consecutive locations $I(k)$ (of nodes in a line graph) as

$$I(k) = \begin{cases} I(k-1)' \circ I(k-2)' \circ \ldots \circ I(0)', & \text{if } k > 0 \\ \{\}, & \text{if } k = 0, \end{cases}$$

where $\circ$ denotes concatenation and for $0 \le j < k$, $I(j)'$ is defined as

$$I(j)' := I(j)^{(1)} \circ i_j^{(1)} \circ I(j)^{(2)} \circ i_j^{(2)} \circ \ldots \circ I(j)^{(c)} \circ i_j^{(c)},$$

where $A^{(\ell)}$ denotes the ℓ^{th} copy of A. Let $N(k)$ be the size of $I(k)$. Since $I(j)'$ consists of c copies of $I(j)$ and a single node i_j, we observe that $N(k)$ satisfies the following recursive relation:

$$\begin{aligned} N(k) &= c(N(k-1)+1) + c(N(k-2)+1) + \cdots + c(N(0)+1) \\ &= c(N(k-1)+1) + N(k-1) \\ &= (c+1)N(k-1) + c, \end{aligned}$$

which implies that $N(k) = \Theta((c+1)^k)$. We have the following pebbling strategy $\mathsf{RevPeb}(I(k))$ for $I(k)$ as shown in Algorithm 1. Here, $\mathsf{RevPeb}^{-1}(I(k))$ denotes the procedure which runs $\mathsf{RevPeb}(I(k))$ in reverse order, i.e., it starts with the final configuration of $\mathsf{RevPeb}(I(k))$ and ends with the starting configuration of $\mathsf{RevPeb}(I(k))$. Intuitively, it sequentially pebbles $I(k-1)', \ldots, I(0)'$ in this order. When we pebble $I(j)' = I(j)^{(1)} \circ i_j^{(1)} \circ I(j)^{(2)} \circ i_j^{(2)} \circ \ldots \circ I(j)^{(c)} \circ i_j^{(c)}$, we run $\mathsf{RevPeb}(I(j)^{(1)})$ to pebble the first block, and we pebble the incident node $i_j^{(1)}$. After that, for $\ell = 2, \ldots, c$, we run $\mathsf{RevPeb}^{-1}(I(j)^{(\ell-1)})$ to remove pebbles from the previous block and we move forward to pebble the next block by running $\mathsf{RevPeb}(I(j)^{(\ell)})$.

Now we have the following lemma.

Algorithm 1: The Procedure $\mathsf{RevPeb}(I(k))$.

Input:
Output:

for $j = k-1, \ldots, 0$ **do**
  Run $\mathsf{RevPeb}(I(j)^{(1)})$
  Pebble node $i_j^{(1)}$
  for $\ell = 2, \ldots, c$ **do**
    Run $\mathsf{RevPeb}^{-1}(I(j)^{(\ell-1)})$
    Run $\mathsf{RevPeb}(I(j)^{(\ell)})$
    Pebble node $i_j^{(\ell)}$
return

Lemma 8. *For a line graph L_N, there exists a reversible pebbling $P \in \mathcal{P}^{\leftrightarrow}_{L_N}$ such that $\Pi^{\leftrightarrow}_{st}(P) = \mathcal{O}\left(N^{1+(2+o(1))\frac{1}{\sqrt{\log N}}}\right)$, and a parallel reversible pebbling $P' \in \mathcal{P}^{\leftrightarrow,\parallel}_{L_N}$ with $\Pi^{\leftrightarrow,\parallel}_{st}(P') = \mathcal{O}\left(N^{1+\frac{2}{\sqrt{\log N}}}\right)$.*

Proof. Let $P = \mathsf{RevPeb}(I(k))$. Then we can easily see that $P \in \mathcal{P}^{\leftrightarrow}_{I(k)}$.

We first consider the space cost of $P = \mathsf{RevPeb}(I(k))$. Intuitively, we first observe that when we pebble $I(k)$, the space cost of pebbling $I(k-1)'$ dominates the space cost of pebbling $I(k-2)', \ldots, I(0)'$ since they are recursively defined. Now when pebbling $I(k-1)'$, we would need to remove pebbles from $I(k-1)^{(\ell)}$ and add pebbles on $I(k-1)^{(\ell+1)}$ for each ℓ, and further, we would need pebbling c intermediate nodes $i_{k-1}^{(1)}, \ldots, i_{k-1}^{(c)}$. Hence, the space complexity of $\mathsf{RevPeb}(I(k))$ satisfies the recurrence relation $\Pi^{\leftrightarrow}_s(\mathsf{RevPeb}(I(k))) \leq \Pi^{\leftrightarrow}_s(\mathsf{RevPeb}(I(k-1))) + c$. Solving the recurrence relation gives us $\Pi^{\leftrightarrow}_s(P) = \mathcal{O}(ck)$.

When it comes to the time cost of P, we would need to be careful and we define $T_f(j)$ to be the amount of time to place a pebble on the last node of $I(j)$, without removing pebbles from earlier nodes in $I(j)$, and we define $T_r(j)$ to be the amount of time to remove such nodes afterwards. Since the pebbling is reversible, we can easily observe that $T_f(j) = T_r(j)$ for each j. In Algorithm 1, when we pebble $I(k)$, we pebble $I(k-1)'$ first which contains the procedure that (1) we pebble $I(k-1)^{(1)}$ and $i_{k-1}^{(1)}$, (2) we remove pebble from $I(k-1)^{(1)}$ and pebble $I(k-1)^{(2)}$, and (3) keep repeating this until the last copy $I(k-1)^{(c)}$ and $i_{k-1}^{(c)}$ is pebbled. Taken together, we have the following recurrence relation for $T_f(k)$:

$$\begin{aligned} T_f(k) &= 2c(T_f(k-1)+1) + T_r(k-1) + \underbrace{2c(T_f(k-2)+1) + T_r(k-2) + \cdots}_{=T_f(k-1)} \\ &= (2c+1)T_f(k-1) + T_r(k-1) + c = (2c+2)T_f(k-1) + c, \end{aligned}$$

which tells us that $T_f(k) = \mathcal{O}\left((2c+2)^k\right)$. Hence, $\Pi^{\leftrightarrow}_t(P) = T_f(k) + T_r(k) = \mathcal{O}\left((2c+2)^k\right)$ and we have $\Pi^{\leftrightarrow}_{st}(P) = \Pi^{\leftrightarrow}_s(P)\Pi^{\leftrightarrow}_t(P) = \mathcal{O}\left(ck(2c+2)^k\right)$.

To express $\Pi^{\leftrightarrow}_{st}(P)$ in terms of $N(k) = \Theta((c+1)^k)$, by setting $c = 2^k$ we observe that

$$\frac{\Pi^{\leftrightarrow}_{st}(P)}{N(k)} = \mathcal{O}\left(ck2^k\right) = \mathcal{O}\left(k4^k\right).$$

We observe that $k4^k = (2^{k^2})^{\frac{2k+\log k}{k^2}} = N(k)^{\frac{2k+\log k}{k^2}}$. Since $N(k) = \Omega(2^{k^2})$ implies $k = \mathcal{O}\left(\sqrt{\log N(k)}\right)$, we have

$$\begin{aligned}\Pi_{st}^{\leftrightarrow}(P) &= \mathcal{O}\left(N(k)\cdot k4^k\right) = \mathcal{O}\left(N(k)^{1+\frac{2k+\log k}{k^2}}\right)\\ &= \mathcal{O}\left(N(k)^{1+\frac{2}{\sqrt{\log N(k)}}+\frac{\log\log N(k)}{2\log N(k)}}\right) = \mathcal{O}\left(N(k)^{1+(2+o(1))\frac{1}{\sqrt{\log N(k)}}}\right).\end{aligned}$$

We can parallelize this strategy by removing pebbles from $I(k-1)^{(\ell)}$ and adding pebbles on $I(k-1)^{(\ell+1)}$ in parallel. If we denote this pebbling strategy $P' = \mathsf{PRevPeb}(I(k))$, then the recurrence relation for the space cost becomes $\Pi_s^{\leftrightarrow,\parallel}(\mathsf{PRevPeb}(I(k))) \le 2\Pi_s^{\leftrightarrow,\parallel}(\mathsf{PRevPeb}(I(k-1))) + c$, which yields $\Pi_s^{\leftrightarrow,\parallel}(P') = \mathcal{O}\left(c2^k\right)$. On the other hand, parallelizing it could save time in each recursion by half, which implies that the recurrence relation for the time cost becomes $T_f(k) = (c+2)T_f(k-1)+c$, which gives us the time cost $\Pi_t^{\leftrightarrow,\parallel}(P') = \mathcal{O}\left((c+2)^k\right)$. In this case, $\Pi_{st}^{\leftrightarrow,\parallel}(P') = \mathcal{O}\left(c(2c+4)^k\right)$. To express $\Pi_{st}^{\leftrightarrow}(P)$ in terms of $N(k) = \Theta((c+1)^k)$, by setting $c+1 = 2^k$ we observe that

$$\begin{aligned}\frac{\Pi_{st}^{\leftrightarrow,\parallel}(P')}{N(k)} &= \mathcal{O}\left(\frac{c(2c+4)^k}{(c+1)^k}\right)\\ &= \mathcal{O}\left(c2^k\left(1+\frac{1}{c+1}\right)^k\right)\\ &= \mathcal{O}\left(2^k\cdot 2^k\cdot 1\right) = \mathcal{O}\left(4^k\right),\end{aligned}$$

since $\left(1+\frac{1}{c+1}\right)^k = \left(1+\frac{1}{2^k}\right)^k = \Theta(1)$[6]. Since $\mathcal{O}\left(4^k\right) = \mathcal{O}\left((2^{k^2})^{2/k}\right) = \mathcal{O}\left(N(k)^{2/k}\right)$ and $N(k) = \Theta(2^{k^2})$ implies $k = \Theta(\sqrt{\log N(k)})$, we have

$$\Pi_{st}^{\leftrightarrow,\parallel}(P') = \mathcal{O}\left(N(k)\cdot 4^k\right) = \mathcal{O}\left(N(k)^{1+\frac{2}{k}}\right) = \mathcal{O}\left(N(k)^{1+\frac{2}{\sqrt{\log N(k)}}}\right). \qquad \square$$

B Reversible Pebbling Strategy Examples

B.1 Example on an (e,d)-Reducible Graph

In this example, we give a DAG $G = (V = [N], E)$ with $N = 16$, and $E = \{(i,i+1) : i \in [15]\} \cup \{((i-1)4+1,(i-1)4+3),((i-1)4+1,(i-1)4+4),((i-1)4+1,(i-1)4+5),((i-1)4+1,(i-1)4+6) : i \in [3]\} \cup \{(13,15),(13,16)\}$, as shown in Figure 2. We observe that G is $(4,3)$-reducible.

Recall that $P = (P_0, P_1, \ldots, P_{2N})$ such that $P_0 = \varnothing$, for $v \in [N], P_v \coloneqq S_{\le v} \cup B_v$, and for $N < v \le 2N$, $P_v \coloneqq P_{2N-v} \cup \{N\}$ is a legal reversible pebbling for G, as shown in Lemma 1, where $B_v \coloneqq \bigcup_{j=1}^{d+1}\bigcup_{i=j}^{d+1}(A_{v+1-j,S,i} \cup A_{v-1+j,S,i})$, with the definition $A_{w,S,i} \coloneqq \left\{v : \mathsf{LongestPath}_{G-S_{\le w-1}}(v,w) = i\right\}$. For example, when we compute P_8 for the graph above, it is described as

$$\begin{aligned}P_8 &= S_{\le 8} \cup B_8\\ &= \{1,5\} \cup \bigcup_{j=1}^{4}\bigcup_{i=j}^{4}(A_{9-j,S,i} \cup A_{7+j,S,i})\end{aligned}$$

[6] for $k > 0$ we have $(1+\frac{1}{2^k})^k < (1+\frac{1}{2^k})^{2^k} < e$.

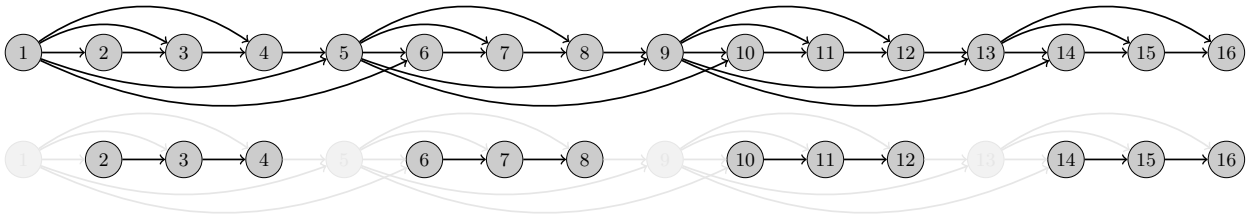

Fig. 2: An (e, d)-reducible DAG G of $N = 4N'$ nodes, with $e = N'$ and $d = 3$ (we set $N' = 4$ in the figure above). Note that with depth-reducing set $S = \{1, 5, 9, 13\}$, we have an original DAG G (top) and the induced subgraph $G - S$ (bottom).

$$
\begin{aligned}
&= \{1,5\} \cup (A_{8,S,1} \cup A_{8,S,2} \cup A_{8,S,3} \cup A_{8,S,4}) \cup (A_{7,S,2} \cup A_{7,S,3} \cup A_{7,S,4} \cup A_{9,S,2} \cup A_{9,S,3} \cup A_{9,S,4}) \\
&\qquad \cup (A_{6,S,3} \cup A_{6,S,4} \cup A_{10,S,3} \cup A_{10,S,4}) \cup (A_{5,S,4} \cup A_{11,S,4}) \\
&= \{1,5\} \cup \{6,7\} \cup \{6,7,8\} \cup \{\} \cup \{2\} \\
&= \{1,2,5,6,7,8\}.
\end{aligned}
$$

Then entire pebbling process is illustrated in Figure 3. Note that for our example, $\Pi_t^{\leftrightarrow,\parallel}(P) = 32 = 2N$ and $\Pi_s^{\leftrightarrow,\parallel}(P) = 9$, which leads to $\Pi_{st}^{\leftrightarrow,\parallel}(P) = 32 \cdot 9 = 288$. While this is not a significant improvement on the naïve pebbling strategy for small $N = 16$, the space-time costs scale with $\mathcal{O}(N)$ for the graphs defined above.

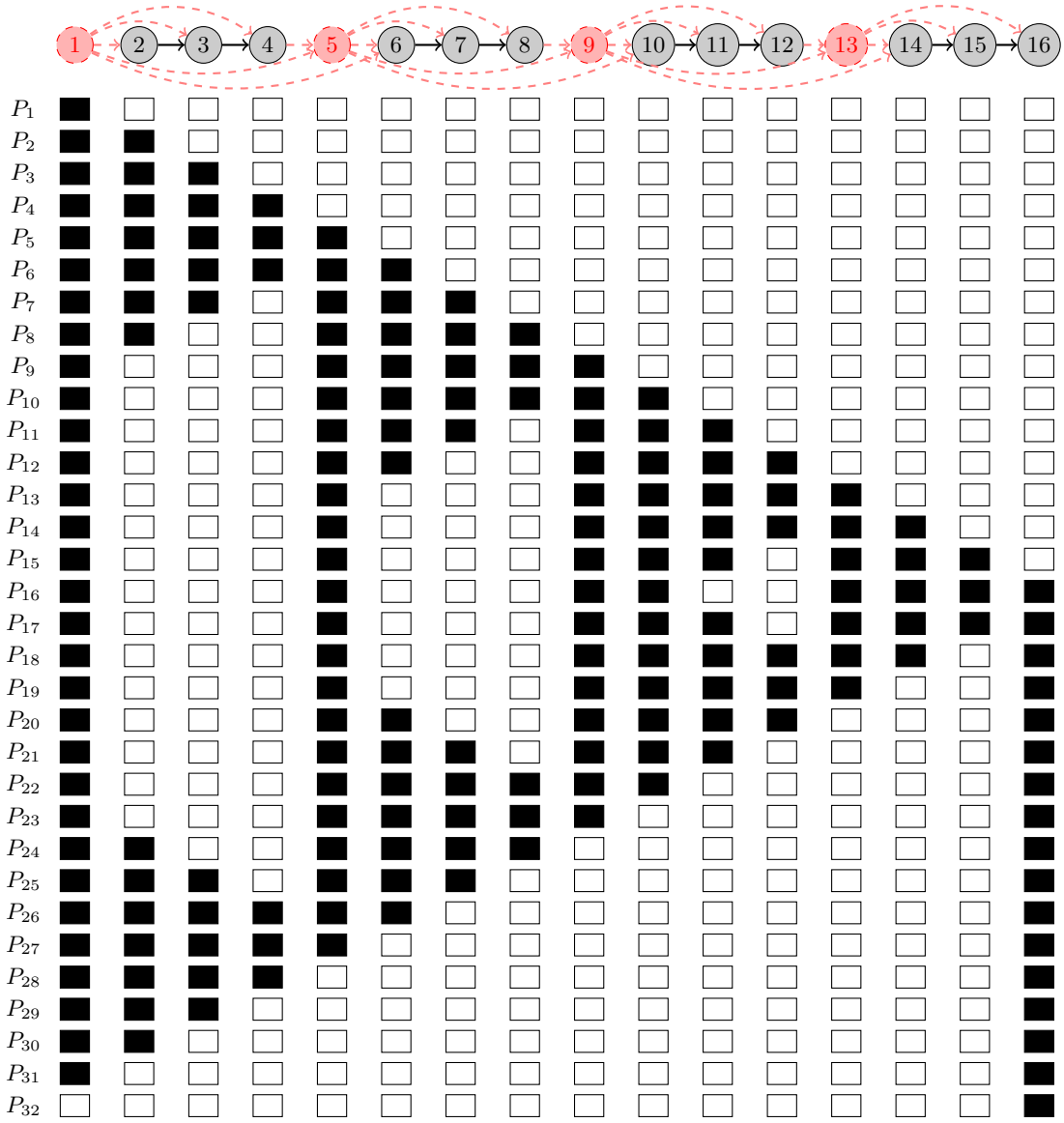

Fig. 3: A parallel reversible pebbling strategy for an (e, d)-reducible graph with $N = 16, e = N/4 = 4$, and $d = 3$. A filled square denotes a pebble on the node and an unfilled square denotes an unpebbled node.

B.2 Example of a Reversible Pebbling Using an Induced Line Graph

In this example, we give a DAG $G = (V = [18], E)$ with the edge distribution as illustrated in Figure 4. As we discussed in Section 3.3, we reduce our DAG G to a line graph L_6 by choosing the block size $b = 3$. Given an efficient reversible pebbling P' of L_6 as shown in Figure 5, we apply $\mathsf{Trans}(G, P', b = 3)$ to produce a legal reversible pebbling of G. Note that we have $\mathsf{LastAdd}(P', 6) = 6$, hence, in reversible pebbling rounds of G that corresponds to P'_6, we pebble all nodes in B_6 and delete pebbles from the block in reverse topological order except for the last node as shown in

Algorithm 2 in Appendix D, which takes $b + N - (\lceil N/b \rceil - 1)b - 1 = 3 + 18 - (6-1)3 - 1 = 5$ steps to complete. We also note that pebbles colored in red are *skip nodes*, which will be kept until the corresponding block is deleted for the last time, i.e., we keep a skip node $v \in B_i$ until we reach rounds that correspond to P'_j (of L_6) with $j = \mathsf{LastDelete}(P', i)$.

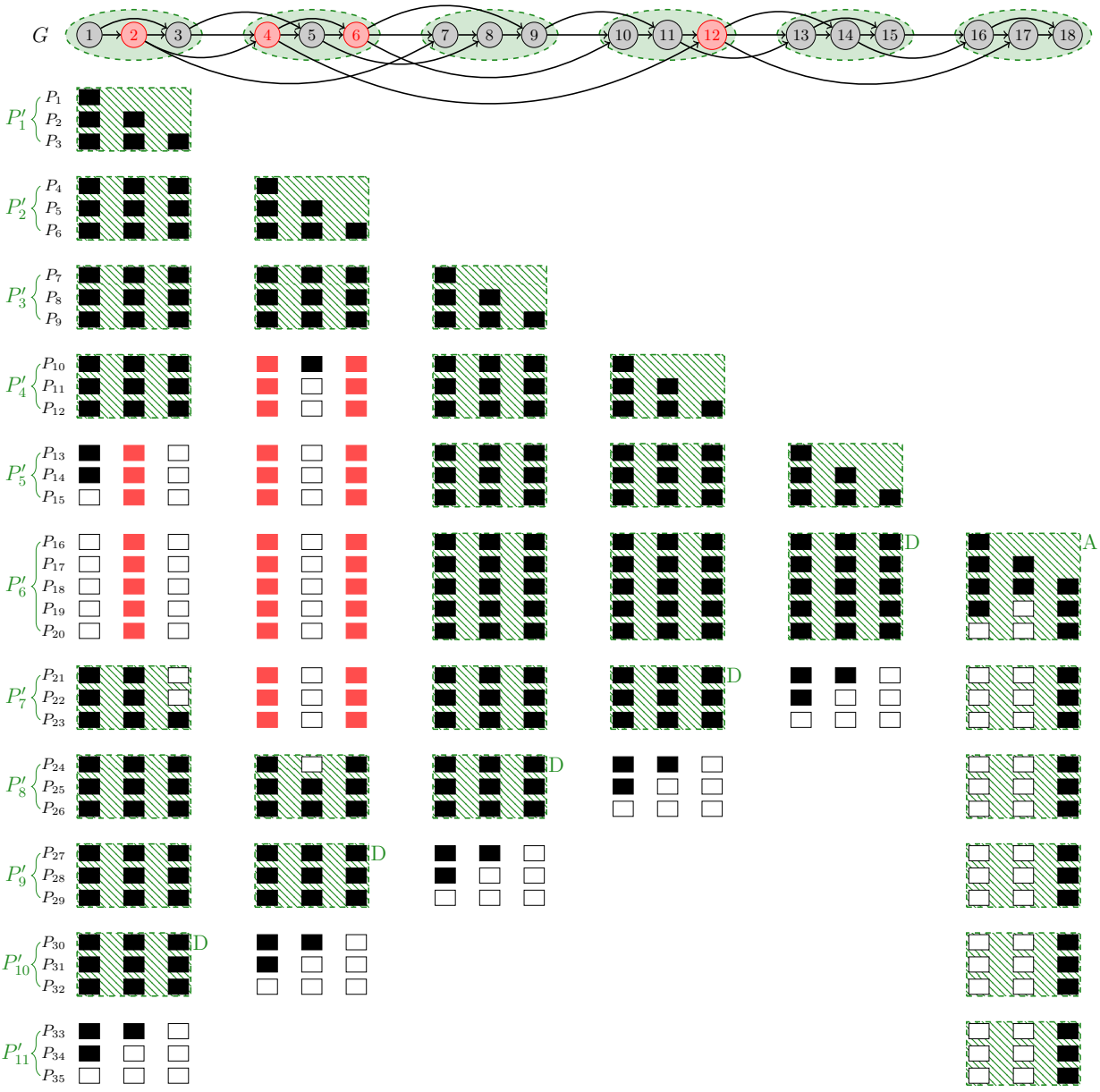

Fig. 4: A parallel reversible pebbling $P = \{P_1, \ldots, P_{35}\}$ of a DAG G using an induced line graph L_6. The (underlying) reversible pebbling for L_6, which is $P' = \{P'_1, \ldots, P'_{11}\}$, is shown in Figure 5. Pebbles colored in red are skip pebbles that cannot be removed until we remove the block of pebbles for the last time, i.e., for each block B_i, we keep pebbles on the skip nodes until we reach P'_j with $j = \mathsf{LastDelete}(P', i)$.

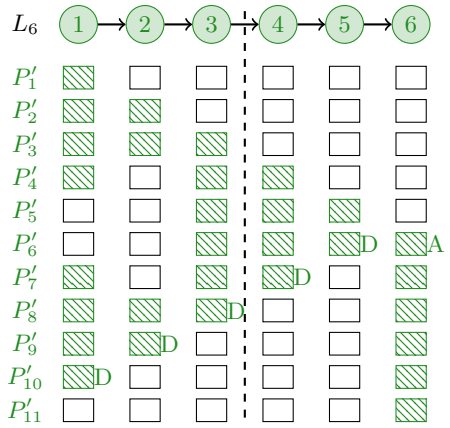

Fig. 5: A reversible pebbling for a line graph with 6 nodes. Note that we mark a pebble on node i in round j with "D" if $j = \mathsf{LastDelete}(P', i)$, and with "A" if $j = \mathsf{LastAdd}(P', i)$.

C Missing Proofs

Reminder of Claim 1. *For $v \in [N]$, $\mathsf{parents}(P_v \setminus P_{v-1}, G) \cup \mathsf{parents}(P_{v-1} \setminus P_v, G) \subseteq P_{v-1} \cap P_v$.*

Proof of Claim 1: We observe that for $v \in [N]$, $P_v \setminus P_{v-1} \subseteq \bigcup_{i=0}^{d} A_{v+i,S,i+1}$, and $P_{v-1} \setminus P_v \subseteq A_{v,S,2} \cup \left(\bigcup_{i=1}^{d+1} A_{v-i,S,i}\right)$. Then by Claim 5 below, we have

$$\begin{aligned}
\mathsf{parents}(P_v \setminus P_{v-1}, G) \setminus S &\subseteq \bigcup_{i=0}^{d} \mathsf{parents}(A_{v+i,S,i+1}, G) \\
&\subseteq \bigcup_{i=0}^{d} A_{v+i,S,i+2} \\
&= \left(\bigcup_{i=0}^{d-1} A_{v+i,S,i+2}\right) \cup A_{v-1+d,S,d+2} \\
&= \bigcup_{i=0}^{d-1} A_{v+i,S,i+2} \subseteq P_{v-1} \cap P_v,
\end{aligned}$$

and

$$\begin{aligned}
\mathsf{parents}(P_{v-1} \setminus P_v, G) \setminus S &\subseteq \mathsf{parents}(A_{v,S,2}, G) \cup \left(\bigcup_{i=1}^{d+1} \mathsf{parents}(A_{v-i,S,i}, G)\right) \\
&\subseteq A_{v,S,3} \cup \left(\bigcup_{i=1}^{d+1} A_{v-i,S,i+1}\right) \\
&= A_{v,S,3} \cup \left(\bigcup_{i=1}^{d} A_{v-i,S,i+1}\right) \cup A_{v-d,S,d+2}
\end{aligned}$$

$$= A_{v,S,3} \cup \left(\bigcup_{i=1}^{d} A_{v-i,S,i+1} \right) \subseteq P_{v-1} \cap P_v,$$

where we have $A_{v-1+d,S,d+2} = A_{w-d,S,d+2} = \varnothing$ by the (e,d)-reducibility. Taken together, we have $\mathsf{parents}(P_v \setminus P_{v-1}, G) \cup \mathsf{parents}(P_{v-1} \setminus P_v, G) \subseteq S_{\leq v-1} \cup (P_{v-1} \cap P_v) = P_{v-1} \cap P_v$. □

Claim 5. $\mathsf{parents}(A_{w,S,i}, G) \setminus S \subseteq A_{w,S,i+1}$.

Proof. If $x \in A_{w,S,i}$ then by definition we have $\mathsf{LongestPath}_{G-S_{\leq w-1}}(x, w) = i$. For any $x' \in \mathsf{parents}(x, G) \setminus S$, we observe that $\mathsf{LongestPath}_{G-S_{\leq w-1}}(x', w) = 1 + \mathsf{LongestPath}_{G-S_{\leq w-1}}(x, w) = i + 1$, which completes the proof. □

Reminder of Lemma 2. *Let $G_{\mathsf{Arg\text{-}A}} = (V_A = [N], E_A)$ and $G_{\mathsf{Arg\text{-}B}} = (V_B = [N], E_B)$ be randomly sampled graphs according to the Argon2i-A and Argon2i-B edge distributions, respectively. Then with high probability, the following holds:*

(1) $G_{\mathsf{Arg\text{-}A}}$ is (e_1, d_1)-reducible for $e_1 = \frac{N}{d'} + \frac{N \ln \lambda}{\lambda}$ and $d_1 = d'\lambda$, for any $0 < \lambda < N$ and $0 < d' < \frac{N}{\lambda}$.
(2) $G_{\mathsf{Arg\text{-}B}}$ is (e_2, d_2)-reducible for $e_2 = \frac{N}{d'} + \frac{2N}{\sqrt{\lambda}}$ and $d_2 = d'\lambda$, for any $0 < \lambda < N$ and $0 < d' < \frac{N}{\lambda}$.

Proof of Lemma 2: We divide N nodes into λ layers of size N/λ and reduce the depth of each layer to d' so that the final depth becomes $d_1 = d_2 = d'\lambda$ for both Argon2i-A and Argon2i-B. To do so, we (a) delete all nodes with parents in the same layer, and (b) delete one out of d' nodes in each layer. Let Delete_i be the event that a node v in i^{th} layer is deleted in step (a), i.e., $r(v)$ remains in the same layer.

(1) For $G_{\mathsf{Arg\text{-}A}}$, since all the layers have the same number of nodes and $r(v)$ is picked uniformly at random from $[v-2]$, we observe that $\Pr[\mathsf{Delete}_i] \leq \frac{1}{i}$. It is clear that we delete N/d' nodes in step (b). Hence,

$$\begin{aligned} e_1 &= \frac{N}{d'} + (\#\text{ nodes deleted in step (a)}) \\ &= \frac{N}{d'} + \sum_{i=1}^{\lambda} \Pr[\mathsf{Delete}_i] \cdot \frac{N}{\lambda} \simeq \frac{N}{d'} + \frac{N \ln \lambda}{\lambda}. \end{aligned}$$

(2) For $G_{\mathsf{Arg\text{-}B}}$, since we have $i\left(1 - \frac{x^2}{N^2}\right) \in (j-1, j]$ if and only if $N\sqrt{1 - \frac{j}{i}} \leq x < N\sqrt{1 - \frac{j-1}{i}}$, we have that $\Pr[r(i) = j] = \sqrt{1 - \frac{j-1}{i}} - \sqrt{1 - \frac{j}{i}}$. Similarly, we have $\Pr[a < r(i) < b] = \Pr_{x \in [N]}\left[i\left(1 - \frac{x^2}{N^2}\right) \in (a, b-1]\right] = \sqrt{1 - \frac{a}{i}} - \sqrt{1 - \frac{b-1}{i}}$. Thus,

$$\begin{aligned} \Pr[\mathsf{Delete}_i] &= \Pr\left[\frac{(i-1)N}{\lambda} < r(v) < v\right] \\ &= \sqrt{1 - \frac{(i-1)N/\lambda}{v}} - \sqrt{1 - \frac{v-1}{v}} \\ &= \sqrt{1 - \frac{(i-1)N}{\lambda v}} - \sqrt{\frac{1}{v}} \end{aligned}$$

$$\leq \sqrt{1-\frac{i-1}{i}} - \sqrt{\frac{\lambda}{iN}} = \sqrt{\frac{1}{i}} - \sqrt{\frac{\lambda}{iN}},$$

where the last inequality holds since $\sqrt{1-\frac{(i-1)N}{\lambda v}} - \sqrt{\frac{1}{v}}$ is an increasing function of v and the largest possible v is iN/λ since it should lie in the i^{th} layer. Hence,

$$\begin{aligned}
e_2 &= \frac{N}{d'} + \sum_{i=1}^{\lambda} \Pr[\mathsf{Delete}_i] \cdot \frac{N}{\lambda} \\
&\leq \frac{N}{d'} + \left(\frac{N}{\lambda} - \sqrt{\frac{N}{\lambda}}\right) \sum_{i=1}^{\lambda} \sqrt{\frac{1}{i}} \\
&\leq \frac{N}{d'} + \left(\frac{N}{\lambda} - \sqrt{\frac{N}{\lambda}}\right) \left(\int_1^{\lambda} \frac{dx}{\sqrt{x}} + 1\right) \\
&= \frac{N}{d'} + \left(\frac{N}{\lambda} - \sqrt{\frac{N}{\lambda}}\right) (2\sqrt{\lambda} - 1) \leq \frac{N}{d'} + \frac{2N}{\sqrt{\lambda}}.
\end{aligned}$$

□

Reminder of Corollary 1. *Let $G_{\mathsf{Arg\text{-}A}} = (V_A = [N], E_A)$ and $G_{\mathsf{Arg\text{-}B}} = (V_B = [N], E_B)$ be randomly sampled graphs according to the Argon2i-A and Argon2i-B edge distributions, respectively. Then with high probability, $\Pi_{st}^{\leftrightarrow,\|}(G_{\mathsf{Arg\text{-}A}}) = \mathcal{O}\left(\frac{N^2 \log\log N}{\sqrt{\log N}}\right)$, and $\Pi_{st}^{\leftrightarrow,\|}(G_{\mathsf{Arg\text{-}B}}) = \mathcal{O}\left(\frac{N^2}{\sqrt[3]{\log N}}\right)$.*

Proof of Corollary 1: From Theorem 2 and Lemma 2, we have

$$\Pi_{st}^{\leftrightarrow,\|}(G_{\mathsf{Arg\text{-}A}}) \leq \mathcal{O}\left(N + Ne + Nd2^d\right) \simeq \mathcal{O}\left(N + \frac{N^2}{d'} + \frac{N^2 \ln\lambda}{\lambda} + \lambda d' 2^{\lambda d'} N\right).$$

To make the upper bound optimal, we want to make the upper bound as small as possible. Hence, we want to find d' and λ such that $\frac{N^2}{d'} \approx \frac{N^2 \ln\lambda}{\lambda} \approx \lambda d' 2^{\lambda d'} N$ as much as possible. Hence, $d' = \frac{\lambda}{\ln\lambda}$ and λ should satisfy $\frac{\lambda^3}{(\ln\lambda)^2} 2^{\lambda^2/\ln\lambda} \approx N$. Setting $\lambda = \sqrt{\log N}$, we have $d' = \frac{\lambda}{\ln\lambda} = \frac{2\sqrt{\log N}}{\ln\log N}$ and $d = d'\lambda = \frac{2\log N}{\ln\log N}$. Thus,

$$\begin{aligned}
\Pi_{st}^{\leftrightarrow,\|}(G_{\mathsf{Arg\text{-}A}}) &\leq \mathcal{O}\left(N + \frac{2N^2 \ln\log N}{2\sqrt{\log N}} + \frac{2N \log N}{\ln\log N} 2^{2\log N/\ln\log N}\right) \\
&= \mathcal{O}\left(N + \frac{2N^2 \ln\log N}{2\sqrt{\log N}} + \frac{2N^{1+\frac{2}{\ln\log N}} \log N}{\ln\log N}\right) \\
&= \mathcal{O}\left(\frac{N^2 \log\log N}{\sqrt{\log N}}\right),
\end{aligned}$$

since $\ln x = (\ln 2)(\log x)$ for any $x > 0$.

For Argon2i-B, we have

$$\Pi_{st}^{\leftrightarrow,\|}(G_{\mathsf{Arg\text{-}B}}) \leq \mathcal{O}\left(N + Ne + Nd2^d\right) \simeq \mathcal{O}\left(N + \frac{N^2}{d'} + \frac{2N^2}{\sqrt{\lambda}} + \lambda d' 2^{\lambda d'} N\right).$$

Similarly, to make the upper bound optimal, we want to make $\frac{N^2}{d'} \approx \frac{2N^2}{\sqrt{\lambda}} \approx \lambda d' 2^{\lambda d'} N$ as much as possible. Hence, we have $d' \approx \sqrt{\lambda}/2$ and plugging in $\lambda = \sqrt[3]{\log^2 N}$ and $d' = \sqrt[3]{\log N}/2$, we have

$$\begin{aligned} \Pi_{st}^{\leftrightarrow,\|}(G_{\mathsf{Arg\text{-}B}}) &\leq \mathcal{O}\left(N + \frac{4N^2}{\sqrt[3]{\log N}} + \frac{N\sqrt{N}\log N}{2}\right) \\ &= \mathcal{O}\left(\frac{N^2}{\sqrt[3]{\log N}}\right). \end{aligned}$$

□

Reminder of Lemma 3. *Let $G = (V = [N], E)$ and $b \in [N]$ be a parameter. If $P' \in \mathcal{P}^{\leftrightarrow,\|}_{L_{\lceil N/b \rceil}}$, then $P = \mathsf{Trans}(G, P', b) \in \mathcal{P}^{\leftrightarrow,\|}_{G}$.*

Proof of Lemma 3: We want to show that it satisfies conditions in Definition 1.

Conditions (1) and (5): $P_{tb+N-(\lceil N/b \rceil - 1)b-1} = \{N\}$.

- It is clear by construction because we remove all nodes except for the target node N.

Condition (2): $\forall j \in [tb + N - (\lceil N/b \rceil - 1)b - 1] : v \in (P_j \setminus P_{j-1}) \Rightarrow \mathsf{parents}(v, G) \subseteq P_{j-1}$.

- We first observe that whenever we pebble a new node w in $L_{\lceil N/b \rceil}$, the node $w-1$ must have been pebbled in the previous round.
- Suppose that $v \in B_w$ for some $w \in [\lceil N/b \rceil]$. For every edge of the form (u, v), we have the following possibilities:
  (a) If $u \in B_w$, u must be (re)pebbled before node v since both u and v corresponds to placing the node w in $L_{\lceil N/b \rceil}$. Hence, $u \in P_{j-1}$.
  (b) If $u \in B_{w-1}$, we are guaranteed that u is already pebbled before we begin pebbling nodes in block B_w since every node in B_{w-1} is pebbled. Hence, $u \in P_{j-1}$.
  (c) If $u \in B_j$ with $j < w - 1$, then u is a skip node and will already be pebbled before placing a pebble on v. Hence, $u \in P_{j-1}$.
- Taken together, we have $\mathsf{parents}(v, G) \subseteq P_{j-1}$.

Condition (3): $\forall j \in [tb + N - (\lceil N/b \rceil - 1)b - 1] : v \in (P_{j-1} \setminus P_j) \Rightarrow \mathsf{parents}(v, G) \subseteq P_{j-1}$.

- We first observe that whenever we remove a pebble from w in $L_{\lceil N/b \rceil}$, the node $w-1$ must have been pebbled in the previous round.
- Suppose that $v \in B_w$ for some $w[\in [\lceil N/b \rceil]$. For every edge of the form (u, v), we have the following possibilities:
  (a) If $u \in B_w$, a pebble on u is not yet removed in the previous round because we remove pebbles in B_w in a reverse topological order. Hence, $u \in P_{j-1}$.
  (b) If $u \in B_{w-1}$, we are guaranteed that u is already pebbled before we begin removing nodes in block B_w since every node in B_{w-1} is pebbled. Hence, $u \in P_{j-1}$.
  (c) If $u \in B_j$ with $j < w-1$, then u is a skip node and will already be pebbled before removing a pebble from v. Hence, $u \in P_{j-1}$.
- Taken together, we have $\mathsf{parents}(v, G) \subseteq P_{j-1}$.

Condition (4): $\forall j \in [tb + N - (\lceil N/b \rceil - 1)b - 1] : v \in \mathsf{parents}(P_j \setminus P_{j-1}, G) \cup \mathsf{parents}(P_{j-1} \setminus P_j, G)$, then $v \in P_j$.

– If $v \in \mathsf{parents}(P_j \setminus P_{j-1}, G)$, then there exists some $v' \in P_j \setminus P_{j-1}$ and some $w \in [\lceil N/b \rceil]$ such that $(v, v') \in E$ and $v' \in B_w$. Now we have the following possibilities:
  (a) If $v \in B_w$, then v must be (re)pebbled before node v' and keep pebbled since both u and v corresponds to placing the node w in $L_{\lceil N/b \rceil}$. Hence, $v \in P_j$.
  (b) If $v \in B_{w-1}$, we are guaranteed that v is already pebbled when we begin pebbling nodes in block B_w since every node in B_{w-1} is pebbled. Hence, $v \in P_j$.
  (c) If $v \in B_j$ with $j < w-1$, then v is a skip node and will already be pebbled and keep pebbled when placing a pebble on v'. Hence, $v \in P_j$.
– If $v \in \mathsf{parents}(P_{j-1} \setminus P_j, G)$, then there exists some $v'' \in P_{j-1} \setminus P_j$ and some $w' \in [\lceil N/b \rceil]$ such that $(v, v'') \in E$ and $v'' \in B_{w'}$. Now we have the following possibilities:
  (a) If $v \in B_{w'}$, a pebble on v is not yet removed in P_j because we remove pebbles in $B_{w'}$ in a reverse topological order. Hence, $v \in P_j$.
  (b) If $v \in B_{w'-1}$, we are guaranteed that v is already pebbled when we begin removing nodes in block $B_{w'}$ since every node in $B_{w'-1}$ is pebbled. Hence, $v \in P_j$.
  (c) If $v \in B_j$ with $j < w' - 1$, then v is a skip node and will already be pebbled and keep pebbled when removing a pebble from v''. Hence, $v \in P_j$.

Taken together, we can conclude that if $P' \in \mathcal{P}^{\leftrightarrow,\parallel}_{L_{\lceil N/b \rceil}}$, then $P = \mathsf{Trans}(G, P', b) \in \mathcal{P}^{\leftrightarrow,\parallel}_{G}$. □

Reminder of Lemma 4. *Let $G_{\mathsf{DRS}} = (V_{\mathsf{DRS}} = [N], E_{\mathsf{DRS}})$ be a randomly sampled graph according to the DRSample edge distribution. Then with high probability, we have* $\mathsf{NumSkip}\left(G_{\mathsf{DRS}}, \left\lceil \frac{N}{\log^2 N} \right\rceil\right) = \mathcal{O}\left(\frac{N \log\log N}{\log N}\right)$.

Proof of Lemma 4: For each $v \in V_{\mathsf{DRS}}$, let Y_v be an indicator random variable for the event that $v - r(v) > b$. Then we observe that $\mathsf{NumSkip}(G_{\mathsf{DRS}}, b) \leq \sum_{v \in V_{\mathsf{DRS}}} Y_v$, since $\mathsf{NumSkip}(G_{\mathsf{DRS}}, b)$ is upper bounded by the number of edges that skip over a block. Since there are at most $\log v$ buckets for $r(v)$ and $\log b$ buckets with $v - r(v) \leq b$, we have $\Pr[v - r(v) > b] \leq 1 - \frac{\log b}{\log v} \leq 1 - \frac{\log b}{\log N} = \frac{\log(N/b)}{\log N}$. Hence, by linearity of expectation it follows that

$$\mathbb{E}[\mathsf{NumSkip}(G_{\mathsf{DRS}}, b)] \leq \sum_{v \in V_{\mathsf{DRS}}} \mathbb{E}[Y_v] = \sum_{v \in V_{\mathsf{DRS}}} \Pr[v - r(v) > b] \leq \sum_{v \in V_{\mathsf{DRS}}} \frac{\log(N/b)}{\log N} = \frac{N \log(N/b)}{\log N}.$$

As the expected value is the sum of independent random variables, we can use Chernoff bounds with $\mu = \frac{N \log(N/b)}{\log N} \geq \sum_{v \in V_{\mathsf{DRS}}} \mathbb{E}[Y_v]$ to show that for any constant $\delta > 0$, we have

$$\Pr[\mathsf{NumSkip}(G_{\mathsf{DRS}}, b) > (1+\delta)\mu] < \exp\left(-\frac{\delta^2 N \log(N/b)}{3 \log N}\right).$$

Hence, with high probability, we have $\mathsf{NumSkip}(G_{\mathsf{DRS}}, b) = \mathcal{O}\left(\frac{N \log(N/b)}{\log N}\right)$. Setting $b = \frac{N}{\log^2 N}$, we get the desired result. □

Reminder of Lemma 6. *Let $\langle P_1, \ldots, P_t \rangle$ and $\langle P'_1, \ldots, P'_{t'} \rangle$ be two legal reversible pebblings for some graph G such that $P_t = P'_{t'}$. Then for any $T \subseteq P_t$,*

$$\langle P_1, \ldots, P_t, P'_{t'-1} \cup T, P'_{t'-2} \cup T, \ldots, P'_1 \cup T \rangle$$

is also a legal reversible pebbling sequence for G.

Proof. First we'll show that $\langle P'_1 \cup T, \ldots, P'_{t'-1} \cup T, P'_{t'} \rangle$ is a legal reversible pebbling. See that since P' satisfies requirements (2), (3), and (4), so does $\langle P'_1 \cup T, \ldots, P'_{t'-1} \cup T, P'_{t'} \rangle$ since no pebbles are removed. Since $P'_t = P'_{t'}$ we have that $\langle P_1, \ldots, P_t, P'_{t'-1} \cup T, P'_{t'-2} \cup T, \ldots, P'_1 \cup T \rangle$ is a legal reversible pebbling sequence. □

Reminder of Lemma 5. *If a legal (non-reversible) pebbling sequence $P = \langle P_1, \ldots, P_t \rangle$ is monotonic, then P is a legal reversible pebbling sequence.*

Proof. Since P is a legal (standard parallel) pebbling and no nodes are deleted, then it suffices to show reversibility. Suppose $x \in \mathsf{parents}(P_i \setminus P_{i-1}) \cup \mathsf{parents}(P_{i-1} \setminus P_i)$. Since

$$\mathsf{parents}(P_i \setminus P_{i-1}) \cup \mathsf{parents}(P_{i-1} \setminus P_i) = \mathsf{parents}(P_i \setminus P_{i-1}),$$

$x \in P_i$ as it was pebbled in some prior pebbling step and P never removes any pebbles. □

Reminder of Lemma 7. *For any (e, d)-depth reducible graph G with depth-reducing set S of size at most e. Then for any $g \in [d, N]$, $P_{rev} = \mathsf{RGenPeb}(G, e, d, S, g)$ is a legal reversible pebbling for G.*

Proof. In the discussion above we've shown that each $\mathcal{P}^c$ and $\mathcal{Q}^c$ are reversible pebblings, so for the first half, it suffices to show that $\langle P_{cg}, Q_1^{c,1} \rangle$ and $\langle Q_1^{c,2}, P_{cg+1} \rangle$ are legal reversible pebblings. Since $Q_1^{c,1} = P_{cg} \cup R(P_{cg})$, it is a legal monotonic (and thus reversible) sequence. Since $Q_0^{c,2} = P_{cg+1}$ (recall we defined Q_d^c but didn't include it in $\mathcal{Q}^c$), $\langle Q_1^{c,2}, P_{cg+1} \rangle$ is also monotonic and thus reversible. Since P^1_{rev} is a reversible pebbling so is P_{rev} by Lemma 6. □

D Reversible Pebbling Strategy using an Induced Line Graph

Algorithm 2: The Procedure $\mathsf{Trans}(G, P', b)$.

Input: A constant-indegree DAG $G = (V = [N], E)$, a parameter b (size of the block), and a legal reversible pebbling $P' = \{P'_0, P'_1, \ldots, P'_t\} \in \mathcal{P}^{\leftrightarrow,\parallel}_{L_{\lceil N/b \rceil}}$ for an induced line graph $L_{\lceil N/b \rceil}$

Output: A legal reversible pebbling $P \in \mathcal{P}^{\leftrightarrow,\parallel}_{G}$ of G

Partition $V = [N]$ into $B_1, \ldots, B_{\lceil N/b \rceil}$ where $B_i = \{(i-1)b+1, (i-1)b+2, \ldots, ib\}$ for $i \in [\lceil N/b \rceil - 1]$ and $B_{\lceil N/b \rceil} = \{(\lceil N/b \rceil - 1)b + 1, (\lceil N/b \rceil - 1)b + 2, \ldots, N\}$.

Initialize $P^{(i)}_{0,b} = \varnothing$ and $P^{(i)}_{j,k} = \varnothing$ for each $i \in [\lceil N/b \rceil]$, $j \in [t]$, and $k \in [f(j)]$, where $f(j) = b + N - \left(\lceil \frac{N}{b} \rceil - 1\right) b - 1$ if $j = \mathsf{LastAdd}(P', \lceil N/b \rceil)$, and $f(j) = b$ elsewhere.

for $i = 1, \ldots, \lceil N/b \rceil - 1$ **do** // for each block $B_1, \ldots, B_{\lceil N/b \rceil - 1}$ except for the last one

  Compute $S_i \coloneqq \mathsf{Skip}(B_i, G)$ using Equation (1).

  for $j = 1, \ldots, t$ **do** // for each round in P'

    if $j \neq \mathsf{LastAdd}(P', \lceil N/b \rceil)$ **then**

      $\{P^{(i)}_{j,1}, \ldots, P^{(i)}_{j,b}\} \leftarrow \mathsf{BlockPebble}(B_i, b, S_i, P', P^{(i)}_{j-1,f(j)}, i, j)$.

    else // i.e., $j = \mathsf{LastAdd}(P', \lceil N/b \rceil)$

      $\{P^{(i)}_{j,1}, \ldots, P^{(i)}_{j,b}\} \leftarrow \mathsf{BlockPebble}(B_i, b, S_i, P', P^{(i)}_{j-1,f(j)}, i, j)$.

      Maintain pebbles for the extra $N - \left(\lceil \frac{N}{b} \rceil - 1\right) b - 1 \leq b - 1$ steps, i.e., $P^{(i)}_{j,b} = P^{(i)}_{j,b+1} = \cdots = P^{(i)}_{j,b+N-\left(\lceil \frac{N}{b} \rceil - 1\right)b-1}$.

for $j = 1, \ldots, t$ **do** // for the last block $B_{\lceil N/b \rceil}$ and for each round in P'

  if $j \neq \mathsf{LastAdd}(P', \lceil N/b \rceil)$ **then**

    $\{P^{(\lceil \frac{N}{b} \rceil)}_{j,1}, \ldots, P^{(\lceil \frac{N}{b} \rceil)}_{j,b}\} \leftarrow \mathsf{LastBlockPebble}(N, b, P', P^{(\lceil \frac{N}{b} \rceil)}_{j-1,f(j)}, j)$.

  else // i.e., $j = \mathsf{LastAdd}(P', \lceil N/b \rceil)$

    $\{P^{(\lceil \frac{N}{b} \rceil)}_{j,1}, \ldots, P^{(\lceil \frac{N}{b} \rceil)}_{j,b}\} \leftarrow \mathsf{LastBlockPebble}(N, b, P', P^{(\lceil \frac{N}{b} \rceil)}_{j-1,f(j)}, j)$.

    Delete pebbles from the block in a reverse topological order, except for the sink node, with $N - (\lceil N/b \rceil - 1)b - 1$ steps, i.e., $P^{(\lceil \frac{N}{b} \rceil)}_{b+k} = P^{(\lceil \frac{N}{b} \rceil)}_{b+k-1} \setminus \{N-k\}$ for $k = 1, \ldots, N - (\lceil N/b \rceil - 1)b - 1$.

for $j = 1, \ldots, t$ **do**

  for $k = 1, \ldots, f(j)$ **do**

    $P_{j,k} = \bigcup_{i=1}^{\lceil N/b \rceil} P^{(i)}_{j,k}$.

    if $j \leq \mathsf{LastAdd}(P', \lceil N/b \rceil)$ **then** // Ordering the pebbling configurations

      $P_{(j-1)b+k} \leftarrow P_{j,k}$

    else

      $P_{N-(\lceil N/b \rceil - 1)b-1+(j-1)b+k} \leftarrow P_{j,k}$

return $P = \{P_1, \ldots, P_{tb+N-(\lceil N/b \rceil - 1)b-1}\}$.

Algorithm 3: The Subfunction $\mathsf{BlockPebble}(B, b, S, P', P_0, i, j)$.

Input: A set of nodes B, a parameter b (size of the set), a set of skip pebbles S, a legal reversible pebbling $P' = \{P'_0, P'_1, \ldots, P'_t\}$, a pebbling configuration P_0 on B, and parameters i and j

Output: A legal relaxed reversible pebbling $P = \{P_1, \ldots, P_b\}$ of the set B

Assert $|B| = b$.
if $i \in P'_j \setminus P'_{j-1}$ **then**
  Place pebbles in the block B with b steps, i.e., $P_1 = P_0 \cup \{(i-1)b+1\}$, and $P_k = P_{k-1} \cup \{(i-1)b+k\}$ for $k = 2, \ldots, b$.
else if $i \in P'_{j-1} \setminus P'_j$ **then**
  if $j - 1 = \mathsf{LastDelete}(P', i)$ **then**
    Delete pebbles from the block B in a reverse topological order with b steps, i.e., $P_1 = P_0 \setminus \{ib\}$, and $P_k = P_{k-1} \setminus \{ib-(k-1)\}$ for $k = 2, \ldots, b$.
  else
    Delete pebbles from the block B *except for the skip nodes*, i.e., $P_1 = P_0 \setminus (\{ib\} \setminus S)$, and $P_k = P_{k-1} \setminus (\{ib-(k-1)\} \setminus S)$ for $k = 2, \ldots, b$.
else
  Maintain pebbles in the block B for b steps, i.e., $P_0 = P_1 = \cdots = P_b$.
return $P = \{P_1, \ldots, P_b\}$

Algorithm 4: The Subfunction $\mathsf{LastBlockPebble}(N, b, P', P_0, j)$.

Input: A parameter N, b, a legal reversible pebbling $P' = \{P'_0, P'_1, \ldots, P'_t\}$, a pebbling configuration P_0 of the last block, and a parameter j

Output: A legal relaxed reversible pebbling $P = \{P_1, \ldots, P_b\}$ of the last block

if $\lceil N/b \rceil \in P'_j \setminus P'_{j-1}$ **then**
  Place pebbles in the block with $N - (\lceil N/b \rceil - 1)b$ steps, and maintain the status for the next $b - N + (\lceil N/b \rceil - 1)b$ steps, i.e., $P_1 = P_0 \cup \{(\lceil N/b \rceil - 1)b + 1\}$, $P_k = P_{k-1} \cup \{(\lceil N/b \rceil - 1)b + k\}$ for $k = 2, \ldots, N - (\lceil N/b \rceil - 1)b$, and $P_{N-(\lceil N/b \rceil-1)b} = P_{N-(\lceil N/b \rceil-1)b+1} = \cdots = P_b$.
else if $\lceil N/b \rceil \in P'_{j-1} \setminus P'_j$ **then**
  Delete pebbles from the block in a reverse topological order with $N - (\lceil N/b \rceil - 1)b$ steps, and maintain the status for the next $b - N + (\lceil N/b \rceil - 1)b$ steps, i.e., $P_1 = P_0 \setminus \{N\}$, $P_k = P_{k-1} \setminus \{N-(k-1)\}$ for $k = 2, \ldots, N - (\lceil N/b \rceil - 1)b$, and $P_{N-(\lceil N/b \rceil-1)b} = P_{N-(\lceil N/b \rceil-1)b+1} = \cdots = P_b$.
else
  Maintain pebbles in the block for b steps, i.e., $P_0 = P_1 = \cdots = P_b$.
return $P = \{P_1, \ldots, P_b\}$