Homework 1 Due Tonight

Advanced Cryptography
CS 655

Week 5:

- Preprocessing: Bit-Fixing Model to Auxiliary Input
- Compression Arguments
- Memory Hard Functions and Pebbling

Spring 2023

Recap: Auxiliary-Input Attacker Model

- **Auxiliary-Input Attacker Model** $A = (A_1, A_2)$
- **Random Oracle Version:**
- Offline attacker A_1 is unbounded and outputs an S-bit hint for online attacker A_2 after viewing entire truth table $H(.)$
- A_2 will try to win security games using this hint

- (S,T,p)-attacker
  - A_1 outputs a S-bit hint
  - A_2 makes at most T random oracle queries
  - A_2 may be constrained in other ways (space/time/signing queries etc…) as specified by parameters p.
- $\left((S,T,p),\varepsilon\right)$-security ➔ Any (S,T,p) attacker wins with advantage at most ε

Recap: Bit-Fixing Model

- **Auxiliary-Input Attacker Model** $A = (A_1, A_2)$
- **Random Oracle Version:**
- Offline attacker A_1 fixes output of random oracle $H(.)$ at P locations and then outputs a S-bit hint.
- A_2 initially knows nothing about remaining unfixed values i.e., $H(x)$ picked randomly for $x \notin P$ after A_1 generates hint
- (P,T,p)-attacker
  - A_1 fixes H on at most P locations and outputs S-bit hint
  - A_2 makes at most T random oracle queries
  - A_2 may be constrained in other ways (space/time/signing queries etc…) as specified by parameters p.
- $\left((S,T,p),\varepsilon\right)$-security ➔ Any (S,T,p) attacker wins with advantage at most ε

Bit-Fixing Model (Unruh)

- **Pro:** Much easier to prove lower bounds in Bit-Fixing Model
- **Con: Bit-Fixing model is n**ot a compelling model for pre-processing attacks

- **Usage:** Lower bound in bit-fixing model ➔ Lower bound in Auxilliary-Input Model

- This approach yields tight lower-bounds in the Auxilliary-Input Model for some applications ☺
- Other applications require a different approach (e.g., compression)

Typical Relationship: BF-RO and AI-RO

Theorem 5. *For any $P \in \mathbb{N}$ and every $\gamma > 0$, if an application G is $((S, T, p), \varepsilon')$-secure in the* BF-RO$(P)$*-model, then it is $((S, T, p), \varepsilon)$-secure in the* AI-RO*-model, for*

$$\varepsilon \ \leq \ \varepsilon' + \frac{2(S + \log \gamma^{-1}) \cdot T_G^{\mathsf{comb}}}{P} + 2\gamma \ ,$$

where T_G^{comb} is the combined query complexity corresponding to G.

Example: Set $\gamma = 2^{-2\lambda}$ and the advantage is $\varepsilon' + \frac{2(S+2\lambda)T}{P} + 2^{-2\lambda}$

Balancing: ε' usually increases with P i.e., as BF-attacker gets to fix more and more points.

Typical Relationship: BF-RO and AI-RO

Theorem 5. *For any $P \in \mathbb{N}$ and every $\gamma > 0$, if an application G is $((S,T,p),\varepsilon')$-secure in the* BF-RO(P)*-model, then it is $((S,T,p),\varepsilon)$-secure in the* AI-RO*-model, for*

$$\varepsilon \;\leq\; \varepsilon' + \frac{2(S + \log\gamma^{-1}) \cdot T_G^{\mathsf{comb}}}{P} + 2\gamma\ ,$$

where T_G^{comb} is the combined query complexity corresponding to G.

So far we have used this result (or similar results for Ideal-Ciphers, Permutations etc…) as a black-box.

How is this result proved?

Preliminary Definitions

- **Definition:** A (N, M)-source is a random variable $\boldsymbol{X}$ with range $[M]^N$
- **Random Oracle $\boldsymbol{H}$**: $[N] \to [M]$ can be viewed as a random variable $\boldsymbol{X}$ with range $[M]^N$ e.g., if $\boldsymbol{H}: \{\mathbf{0}, \mathbf{1}\}^{\boldsymbol{n}} \to \{\mathbf{0}, \mathbf{1}\}^{\boldsymbol{m}}$ then we set $M = 2^m$ and $N = 2^n$
- Given $I \subseteq [N]$ (inputs) and $x \in [M]^N$ let $x_I \in [M]^{|I|}$ denote the substring specified by I e.g., value of random oracle on all inputs in I
- **Dense-Source:** $\boldsymbol{X}$ is $(1-\delta)$ dense if for **every** subset $I \subseteq [N]$ (inputs) we have $H_\infty(X_I) \geq (1-\delta)|I| \log_2 M = (1-\delta) \log_2 M^{|I|}$

Minimum Entropy: Equivalent statement is that for all $y \in [M]^{|I|}$ we have $\Pr[X_I = y] \leq |M|^{-|I|(1-\delta)}$

Preliminary Definitions

- **Definition:** A (N, M)**-**source is a random variable $\boldsymbol{X}$ with range $[M]^N$

- **Random Oracle** $\boldsymbol{H}$: $[N] \rightarrow [M]$ can be viewed as a random variable $\boldsymbol{X}$ with range $[M]^N$ e.g., if $\boldsymbol{H}$: $\{\mathbf{0}, \mathbf{1}\}^{\boldsymbol{n}} \rightarrow \{\mathbf{0}, \mathbf{1}\}^{\boldsymbol{m}}$ then we set $M = 2^m$ and $N = 2^n$

- **Dense-Source:** $\boldsymbol{X}$ is $(1 - \delta)$ $-$dense if for **every** subset $I \subseteq [N]$ (inputs) we have $H_\infty(X_I) \geq (1 - \delta)|I| \log_2 M = (1 - \delta) \log_2 M^{|I|}$
- **Example:** Random oracle is $(1 - \delta)$ $-$dense with $\delta = 0$.

Preliminary Definitions

- **Definition:** A (N, M)-source is a random variable $\boldsymbol{X}$ with range $[M]^N$

- **Dense-Source:** $\boldsymbol{X}$ is $(P, 1-\delta)$ −dense if there is a subset $S \subseteq [N]$ of size $|S| \leq P$ such that for **every** subset $I \subseteq [N \setminus \boldsymbol{S}]$ we have
$$H_\infty(X_I) \geq (1-\delta)|I| \log_2 M = (1-\delta) \log_2 M^{|I|}$$

- **Intuition:** Fixed on P coordinates but dense on the rest

- **Bit-Fixing Source:** $\boldsymbol{X}$ is $(P, 1)$ −dense i.e., fixed on P and uniform on the rest

Preliminary: Leaky Source

- **Leaky-Source (Auxiliary-Input):** Online attacker gets hint $\mathrm{z} = \mathrm{f}(\boldsymbol{X})$ for some function f: $[M]^N \rightarrow \{\mathbf{0}, \mathbf{1}\}^{\boldsymbol{S}}$.
  - **Bayesian Update:** Conditional Distribution $\boldsymbol{X_z}$ *for all x in* $[M]^N$ *we have*
$$\Pr[\boldsymbol{X_z} = x] \coloneqq \Pr[X = x | f(X) = z]$$

  Challenge: It can be difficult to reason about the source $\boldsymbol{X_z}$

  Entropy Deficiency: $S_z = N \log M - H_\infty(\boldsymbol{X_z})$

  In expectation we have $\mathbf{E}[S_z] \le S$, but the actual value can vary depending on $\mathrm{z} = \mathrm{f}(\boldsymbol{X})$

Proof Strategy

- **Leaky-Source (Auxiliary-Input):** Online attacker gets hint $\mathrm{z} = \mathrm{f}(\boldsymbol{X})$ for some function f: $[M]^N \rightarrow \{\mathbf{0}, \mathbf{1}\}^{\boldsymbol{S}}$.
  - **Bayesian Update:** Conditional Distribution $\boldsymbol{X_z}$ *for all x in* $[M]^N$ *we have*
    $$\Pr[\boldsymbol{X_z} = x] \coloneqq \Pr[X = x | f(X) = z]$$
  Challenge: It can be difficult to reason about the source $\boldsymbol{X_z}$
  Entropy Deficiency: $S_z = N \log M - H_\infty(\boldsymbol{X_z})$ $\quad\quad\quad \mathbf{E}[S_z] \le S$

- **Step 1:** Show that any leaky source $\boldsymbol{X_z}$ is γ-close to a source $\boldsymbol{Y_z}$ which is a convex combination of $(P', 1-\delta)$-dense sources.
  - **Convex Combination:** Let $D_1, \dots, D_k$ each be $(P', 1-\delta)$-dense sources. $\boldsymbol{Y_z}$ has the form sample a source $i \le k$ with probability p_i then sample from $(P', 1-\delta)$-dense sources D_i
  - **Number of Fixed Points:** $P' \le \frac{S_Z + \log 1/\gamma}{\delta \log M}$

Proof Strategy

- **Leaky-Source (Auxiliary-Input):** Online attacker gets hint $\mathrm{z} = \mathrm{f}(\boldsymbol{X})$ for some function f: $[M]^N \rightarrow \{\mathbf{0}, \mathbf{1}\}^{\boldsymbol{S}}$.
  - **Bayesian Update:** Conditional Distribution $\boldsymbol{X_z}$ *for all x in* $[M]^N$ *we have*

$$\Pr[\boldsymbol{X_z} = x] \coloneqq \Pr[X = x | f(X) = z]$$

  Challenge: It can be difficult to reason about the source $\boldsymbol{X_z}$

  Entropy Deficiency: $S_z = N \log M - H_\infty(\boldsymbol{X_z})$ $\mathbf{E}[S_z] \leq S$

- **Step 2:** Show that a $(P', 1-\delta)$-dense source X' cannot be distinguished from a P'-bit fixing source Y' by a distinguisher making at most T (adaptive) queries.

$$\Pr\left[\mathfrak{D}^{X'} = 1\right] \leq \Pr\left[\mathfrak{D}^{Y'} = 1\right] \times M^{T\delta}$$

  And

$$\left|\Pr\left[\mathfrak{D}^{X'} = 1\right] - \Pr\left[\mathfrak{D}^{Y'} = 1\right]\right| \leq T\delta \log M$$

Proof Strategy

- **Step 1:** Show that any leaky source $\boldsymbol{X_z}$ is γ-close to a source $\boldsymbol{Y_z}$ which is a convex combination of $(P', 1-\delta)$-dense sources.

- Define $\mathbf{Y} = \boldsymbol{X_z}$. If $\mathbf{Y}$ is $(1-\delta)$-dense then we are done. Otherwise, let I be the largest subset for which there exists a violation i.e., $y_I \in [M]^{|I|}$ s.t.
$$\Pr[\boldsymbol{Y_I} = y_I] > 2^{-(1-\delta)|I| \log M}$$
- Let $\mathbf{Y}'$ denote distribution of $\mathbf{Y}$ conditioned on $\boldsymbol{Y_I} = y_I$
- **Claim 1:** $\mathbf{Y}'$ is $(P', 1-\delta)$ dense with $P' = |I|$
  - **Proof Sketch:** If there is a subset $\boldsymbol{J} \subseteq [\boldsymbol{N} \backslash \mathrm{I}]$ and $y_J \in [M]^{|J|}$ s.t.
  $$\Pr\left[\boldsymbol{Y_J}' = y_j\right] > 2^{-(1-\delta)|J| \log M}$$
  Then we could take $I' = I \cup J$ and
  $$\Pr[\boldsymbol{Y_{I'}} = y_{I'}] = \Pr[\boldsymbol{Y_I} = y_I] \Pr\left[\boldsymbol{Y_J} = y_J | \boldsymbol{Y_I} = y_I\right] > 2^{-(1-\delta)|I'| \log M}$$
  This contradicts the maximality of I!

Proof Strategy

- **Step 1:** Show that any leaky source $\boldsymbol{X_Z}$ is γ-close to a source $\boldsymbol{Y_Z}$ which is a convex combination of $(P', 1-\delta)$-dense sources.

- Define $\mathbf{Y} = \boldsymbol{X_Z}$. If $\mathbf{Y}$ is $(1-\delta)$-dense then we are done. Otherwise, let I be the largest subset for which there exists a violation i.e., $y_I \in [M]^{|I|}$ s.t.
$$\Pr[\boldsymbol{Y_I} = y_I] > 2^{-(1-\delta)|I| \log M}$$
- Let $\mathbf{Y}'$ denote distribution of $\mathbf{Y}$ conditioned on $\boldsymbol{Y_I} = y_I$
- **Claim 1:** $\mathbf{Y}'$ is $(P', 1-\delta)$ dense with $P' = |I|$
- **Claim 2:** $|I| \leq \frac{S_Z}{(\delta \log M)}$
  - **Proof Sketch:** On one hand we have $H_\infty(\boldsymbol{Y_I}) \geq |I| \log M - S_Z$ (def of S_Z)
  - On the other hand $H_\infty(\boldsymbol{Y_I}) < -\log_{\mathbf{2}}\left(2^{-(1-\delta)|I| \log M}\right) = (1-\delta)|I| \log M$
  - Claim 2 follows immediately by combining the above two inequalities.

Proof Strategy

- **Step 1:** Show that any leaky source $\boldsymbol{X_z}$ is γ-close to a source $\boldsymbol{Y_z}$ which is a convex combination of $(P', 1-\delta)$-dense sources.

- Define $\mathbf{Y} = \boldsymbol{X_z}$. If $\mathbf{Y}$ is $(1-\delta)$-dense then we are done. Otherwise, let I be the largest subset for which there exists a violation i.e., $y_I \in [M]^{|I|}$ s.t.
$$\Pr[\boldsymbol{Y_I} = y_I] > 2^{-(1-\delta)|I| \log M}$$
- Let $\mathbf{Y}'$ denote distribution of $\mathbf{Y}$ conditioned on $\boldsymbol{Y_I} = y_I$
- **Claim:** $\mathbf{Y}'$ is $(P', 1-\delta)$ dense with $P' = |I|$ with $|I| \leq \frac{S_Z}{(\delta \log M)}$
- **Key Idea (Recursion!):** $\boldsymbol{Y_z}$ uses $(P', 1-\delta)$ dense source $\mathbf{Y}'$ with probability $\Pr[\boldsymbol{Y_I} = y_I]$ and samples from $\boldsymbol{Y_z}'$ with probability $1 - \Pr[\boldsymbol{Y_I} = y_I]$
- $\boldsymbol{Y_z}'$ is also convex combination of finitely many $(P', 1-\delta)$-dense sources which is gamma close to $\boldsymbol{Y_1}$, the distribution of $\mathbf{Y}$ conditioned on $\boldsymbol{Y_I} \neq y_I$

Proof Strategy

- **Step 1:** Show that any leaky source $\boldsymbol{X_Z}$ is γ-close to a source $\boldsymbol{Y_Z}$ which is a convex combination of $(P', 1-\delta)$-dense sources.

- **Key Idea (Recursion!):** $\boldsymbol{Y_Z}$ uses $(P', 1-\delta)$ dense source $\mathbf{Y}'$ with probability $\Pr[\boldsymbol{Y_I} = y_I]$ and samples from $\boldsymbol{Y_Z}'$ with probability $1 - \Pr[\boldsymbol{Y_I} = y_I]$
- $\boldsymbol{Y_Z}'$ is also convex combination of finitely many $(P', 1-\delta)$-dense sources which is gamma close to $\boldsymbol{Y_1}$, the distribution of $\mathbf{Y}$ conditioned on $\boldsymbol{Y_I} \neq y_I$
- Each step of recursion decreases size of support ➔ finite termination
- Recurse as long as $\Pr[X \in \mathrm{Supp}(Y_k)] > \gamma$
- **Claim:** $\boldsymbol{Y_k}'$ is $(P', 1-\delta)$ dense with $P' \leq \frac{S_Z + \log\frac{1}{\gamma}}{(\delta \log M)}$
- Process ends with $\Pr\left[X \in \mathrm{Supp}\left(Y_{final}\right)\right] \leq \gamma$ ➔ replace Y_{final} with uniform distribution

Proof Strategy

- **Step 2:** Show that a $(P', 1-\delta)$-dense source X' cannot be distinguished from corresponding P'-bit fixing source Y' (uniform on non-fixed coordinates) by a distinguisher making at most T (adaptive) queries to the source.

$$\Pr\left[\mathfrak{D}^{X'} = 1\right] \leq \Pr\left[\mathfrak{D}^{Y'} = 1\right] \times M^{T\delta}$$

And

$$\left|\Pr\left[\mathfrak{D}^{X'} = 1\right] - \Pr\left[\mathfrak{D}^{Y'} = 1\right]\right| \leq T\delta \log M$$

Claim 3. *For any $(P', 1-\delta)$-dense source X' and its corresponding P'-bit-fixing source Y', it holds that for any (adaptive) distinguisher $\mathcal{D}$ that queries at most T coordinates of its oracle,*

$$\left|\mathsf{P}\left[\mathcal{D}^{X'} = 1\right] - \mathsf{P}\left[\mathcal{D}^{Y'} = 1\right]\right| \ \leq \ T\delta \cdot \log M,$$

and

$$\mathsf{P}\left[\mathcal{D}^{X'} = 1\right] \ \leq \ M^{T\delta} \cdot \mathsf{P}\left[\mathcal{D}^{Y'} = 1\right].$$

Proof Strategy

- **Step 2:** Show that a $(P', 1-\delta)$-dense source X’ cannot be distinguished from a P'-bit fixing source Y’ by a distinguisher making at most T (adaptive) queries to the source.

$$\Pr\left[\mathfrak{D}^{X'} = 1\right] \leq \Pr\left[\mathfrak{D}^{Y'} = 1\right] \times M^{T\delta}$$

And

$$\left|\Pr\left[\mathfrak{D}^{X'} = 1\right] - \Pr\left[\mathfrak{D}^{Y'} = 1\right]\right| \leq T\delta \log M$$

Proof Intuition:

- WLOG we can assume $\mathfrak{D}$ is deterministic (otherwise we can fix the random coins that maximizes the advantage of the distinguisher for $\mathfrak{D}$) and only queries on non-fixed points.
- Transcript τ is a list of all of the query/answer pairs that distinguisher $\mathfrak{D}$ makes.

Proof Strategy

- **Step 2:** Show that a $(P', 1-\delta)$-dense source X' cannot be distinguished from a P'-bit fixing source Y' by a distinguisher making at most T (adaptive) queries to the source.

$$\Pr\left[\mathfrak{D}^{X'} = 1\right] \leq \Pr\left[\mathfrak{D}^{Y'} = 1\right] \times M^{T\delta}$$

And

$$\left|\Pr\left[\mathfrak{D}^{X'} = 1\right] - \Pr\left[\mathfrak{D}^{Y'} = 1\right]\right| \leq T\delta \log M$$

Proof Intuition: Transcript τ is a list of all of the query/answer pairs that distinguisher $\mathfrak{D}$ makes.

- Let $T_{X'}$ (resp. $T_{Y'}$) denote random variable over transcripts resulting from interaction with source X' (resp. Y').
- Note: The support of $T_{Y'}$ contains the support of $T_{X'}$
- For every transcript τ in the support of $T_{X'}$ we have

$$\Pr[T_{X'} = \tau] \leq 2^{-(1-\delta)T \log M} \quad and \quad \Pr[T_{Y'} = \tau] = 2^{-T \log M}$$

$$\Pr[T_{X'} = \tau] \leq M^{T\delta} \Pr[T_{Y'} = \tau]$$

Proof Strategy

- **Step 2:** Show that a $(P', 1-\delta)$-dense source X' cannot be distinguished from a P'-bit fixing source Y' by a distinguisher making at most T (adaptive) queries to the source.

$$\Pr\left[\mathfrak{D}^{X'} = 1\right] \leq M^{T\delta} \times \Pr\left[\mathfrak{D}^{Y'} = 1\right]$$

And

$$\left|\Pr\left[\mathfrak{D}^{X'} = 1\right] - \Pr\left[\mathfrak{D}^{Y'} = 1\right]\right| \leq T\delta \log M$$

Proof Intuition: For every transcript τ in the support of T_Y, we have

$$\Pr[T_{X'} = \tau] \leq 2^{-(1-\delta)T \log M} \quad and \quad \Pr[T_{Y'} = \tau] = 2^{-T \log M}$$

$$\Pr[T_{X'} = \tau] \leq M^{T\delta} \Pr[T_{Y'} = \tau]$$

Let $\mathcal{T}_{\mathfrak{D}}$ denote the set of all transcripts where $\mathfrak{D}$ outputs 1.

$$\Pr\left[\mathfrak{D}^{X'} = 1\right] = \sum_{\tau \in \mathcal{T}_{\mathfrak{D}}} \Pr[T_{X'} = \tau] \leq M^{T\delta} \sum_{\tau \in \mathcal{T}_{\mathfrak{D}}} \Pr[T_{Y'} = \tau] = M^{T\delta} \times \Pr\left[\mathfrak{D}^{Y'} = 1\right]$$

Claim 3. *For any $(P', 1-\delta)$-dense source X' and its corresponding P'-bit-fixing source Y', it holds that for any (adaptive) distinguisher $\mathcal{D}$ that queries at most T coordinates of its oracle,*

$$\left|\mathsf{P}\left[\mathcal{D}^{X'}=1\right]-\mathsf{P}\left[\mathcal{D}^{Y'}=1\right]\right| \leq T\delta\cdot\log M,$$

and

and also $\mathsf{P}[T_{Y'}=\tau]=\mathsf{p}_{Y'}(\tau)$. Towards proving the first part of the lemma, observe that

and a

$$\begin{aligned}\left|\mathsf{P}\left[\mathcal{D}^{X'}=1\right]-\mathsf{P}\left[\mathcal{D}^{Y'}=1\right]\right| &\leq \mathsf{SD}(T_{X'},T_{Y'})\\ &= \sum_{\tau}\max\left\{0,\mathsf{P}[T_{X'}=\tau]-\mathsf{P}[T_{Y'}=\tau]\right\}\\ &= \sum_{\tau\in\mathcal{T}_X}\max\left\{0,\mathsf{p}_{X'}(\tau)-\mathsf{p}_{Y'}(\tau)\right\}\\ &= \sum_{\tau\in\mathcal{T}_X}\mathsf{p}_{X'}(\tau)\cdot\max\left\{0,1-\frac{\mathsf{p}_{Y'}(\tau)}{\mathsf{p}_{X'}(\tau)}\right\}\\ &\leq 1-M^{-T\delta} \;\leq\; T\delta\cdot\log M,\end{aligned}$$

where the first sum is over all possible transcripts and where the last inequality uses $2^{-x}\geq 1-x$ for $x\geq 0$.

$$\leq 1-M^{-T\delta} \;\leq\; T\delta\cdot\log M,$$

where the first sum is over all possible transcripts and where the last inequality uses $2^{-x}\geq 1-x$ for $x\geq 0$.

Lemma 1. *Let X be distributed uniformly over $[M]^N$ and $Z := f(X)$, where $f : [M]^N \to \{0,1\}^S$ is an arbitrary function. For any $\gamma > 0$ and $P \in \mathbb{N}$, there exists a family $\{Y_z\}_{z\in\{0,1\}^S}$ of convex combinations Y_z of P-bit-fixing (N,M)-sources such that for any distinguisher D taking an S-bit input and querying at most $T < P$ coordinates of its oracle,*

$$\left|\mathsf{P}\big[\mathcal{D}^X(f(X))=1\big]-\mathsf{P}\big[\mathcal{D}^{Y_{f(X)}}(f(X))=1\big]\right| \;\le\; \frac{(S+\log 1/\gamma)\cdot T}{P}+\gamma$$

and

$$\mathsf{P}\big[\mathcal{D}^X(f(X))=1\big] \;\le\; 2^{(S+2\log 1/\gamma)T/P}\cdot\mathsf{P}\big[\mathcal{D}^{Y_{f(X)}}(f(X))=1\big]+2\gamma\,.$$

Let Y_z' be obtained by replacing every X' by the corresponding Y' in X_z'. Setting $\delta_z = (S_z + \log 1/\gamma)/(P\log M)$, Claims 2 and 3 imply

$$\left|\mathsf{P}\big[\mathcal{D}^{X_z}(z)=1\big]-\mathsf{P}\big[\mathcal{D}^{Y_z'}(z)=1\big]\right| \;\le\; \frac{(S_z+\log 1/\gamma)\cdot T}{P}+\gamma\,, \tag{2}$$

as well as

$$\mathsf{P}\big[\mathcal{D}^{X_z}(z)=1\big] \;\le\; 2^{(S_z+\log 1/\gamma)T/P}\cdot\mathsf{P}\big[\mathcal{D}^{Y_z'}(z)=1\big]+\gamma\,. \tag{3}$$

Moreover, note that for the above choice of δ_z, $P' = P$, i.e., the sources Y' are fixed on at most P coordinates, as desired.

Lemma 1. *Let X be distributed uniformly over $[M]^N$ and $Z := f(X)$, where $f : [M]^N \to \{0,1\}^S$ is an arbitrary function. For any $\gamma > 0$ and $P \in \mathbb{N}$, there exists a family $\{Y_z\}_{z \in \{0,1\}^S}$ of convex combinations Y_z of P-bit-fixing (N, M)-sources such that for any distinguisher D taking an S-bit input and querying at most $T < P$ coordinates of its oracle,*

$$\left|\mathsf{P}\left[\mathcal{D}^X(f(X)) = 1\right] - \mathsf{P}\left[\mathcal{D}^{Y_{f(X)}}(f(X)) = 1\right]\right| \leq \frac{(S + \log 1/\gamma) \cdot T}{P} + \gamma$$

and

$$\mathsf{P}\left[\mathcal{D}^X(f(X)) = 1\right] \leq 2^{(S+2\log 1/\gamma)T/P} \cdot \mathsf{P}\left[\mathcal{D}^{Y_{f(X)}}(f(X)) = 1\right] + 2\gamma .$$

One Missing Link Remains! Prior bounds relied on entropy deficiency $S_z = N \log M - H_\infty(\boldsymbol{X_z})$ instead of S.

Claim: $E[S_z] \leq S$ and $\Pr\left[S_{f(X)} > S + \log\frac{1}{\gamma}\right] \leq \gamma$

Key Fact: $\Pr[f(X) = z] \leq 2^{-S_z}$

Lemma 1. *Let X be distributed uniformly over $[M]^N$ and $Z := f(X)$, where $f : [M]^N \to \{0,1\}^S$ is an arbitrary function. For any $\gamma > 0$ and $P \in \mathbb{N}$, there exists a family $\{Y_z\}_{z \in \{0,1\}^S}$ of convex combinations Y_z of P-bit-fixing (N, M)-sources such that for any distinguisher D taking an S-bit input and querying at most $T < P$ coordinates of its oracle,*

$$\left|\mathsf{P}\left[\mathcal{D}^X(f(X)) = 1\right] - \mathsf{P}\left[\mathcal{D}^{Y_{f(X)}}(f(X)) = 1\right]\right| \le \frac{(S + \log 1/\gamma) \cdot T}{P} + \gamma$$

and

$$\mathsf{P}\left[\mathcal{D}^X(f(X)) = 1\right] \le 2^{(S+2\log 1/\gamma)T/P} \cdot \mathsf{P}\left[\mathcal{D}^{Y_{f(X)}}(f(X)) = 1\right] + 2\gamma .$$

Claim: $E[S_z] \le S$ and $\Pr\left[S_{f(X)} > S + \log\frac{1}{\gamma}\right] \le \gamma$

Key Fact: $\Pr[f(X) = z] \le 2^{-S_z}$

Proof: By definition of S_z (min-entropy deficiency) there exists $x \in [M]^N$ with $f(X) = z$ such that $\Pr[X = x | f(X) = z] = \frac{2^{S_z}}{M^N}$. We have

$$\frac{1}{M^N} = \Pr[X = x] = \Pr[X = x | f(X) = z] \Pr[f(X) = z] = \frac{2^{S_z}}{M^N} \Pr[f(X) = z]$$

Lemma 1. *Let X be distributed uniformly over $[M]^N$ and $Z := f(X)$, where $f : [M]^N \to \{0,1\}^S$ is an arbitrary function. For any $\gamma > 0$ and $P \in \mathbb{N}$, there exists a family $\{Y_z\}_{z\in\{0,1\}^S}$ of convex combinations Y_z of P-bit-fixing (N,M)-sources such that for any distinguisher D taking an S-bit input and querying at most $T < P$ coordinates of its oracle,*

$$\left|\mathsf{P}\left[\mathcal{D}^X(f(X)) = 1\right] - \mathsf{P}\left[\mathcal{D}^{Y_{f(X)}}(f(X)) = 1\right]\right| \;\le\; \frac{(S + \log 1/\gamma)\cdot T}{P} + \gamma$$

and

$$\mathsf{P}\left[\mathcal{D}^X(f(X)) = 1\right] \;\le\; 2^{(S+2\log 1/\gamma)T/P} \cdot \mathsf{P}\left[\mathcal{D}^{Y_{f(X)}}(f(X)) = 1\right] + 2\gamma\,.$$

Claim: $E[S_z] \le S$ and $\Pr\left[S_{f(X)} > S + \log\frac{1}{\gamma}\right] \le \gamma$

Key Fact: $\Pr[f(X) = z] \le 2^{-S_z}$

$$\Pr\left[S_{f(X)} > S + \log\frac{1}{\gamma}\right] = \sum_{\substack{z\in\{0,1\}^S \ s.t \\ S_z > S + \log\frac{1}{\gamma}}} \Pr[f(X) = z] \le 2^S \times 2^{-\left(S + \log\frac{1}{\gamma}\right)} \le \gamma$$

Claim 4. $\mathbf{E}_z[S_z] \leq S$ *and* $\mathsf{P}\big[S_{f(X)} > S + \log 1/\gamma\big] \leq \gamma$.

Proof. Observe that $H_\infty(X_z) = H_\infty(X|Z = z) = H(X|Z = z)$ since, conditioned on $Z = z$, X is distributed uniformly over all values x with $f(x) = z$. Therefore,

$$\begin{aligned} \mathbf{E}_z[S_z] &= N \log M - \mathbf{E}_z[H_\infty(X|Z = z)] = N \log M - \mathbf{E}_z[H(X|Z = z)] \\ &= N \log M - H(X|Z) \leq S . \end{aligned}$$

Again due to the uniformity of X, $\mathsf{P}[f(X) = z] = 2^{-S_z}$. Hence,

$$\mathsf{P}\big[S_{f(X)} > S + \log 1/\gamma\big] = \sum_{z \in \{0,1\}^S :\, S_z > S + \log 1/\gamma} \mathsf{P}\big[f(X) = z\big] \leq 2^S \cdot 2^{-(S + \log 1/\gamma)} \leq \gamma .$$

$\square$

Lemma 1. *Let X be distributed uniformly over $[M]^N$ and $Z := f(X)$, where $f : [M]^N \to \{0,1\}^S$ is an arbitrary function. For any $\gamma > 0$ and $P \in \mathbb{N}$, there exists a family $\{Y_z\}_{z \in \{0,1\}^S}$ of convex combinations Y_z of P-bit-fixing (N, M)-sources such that for any distinguisher D taking an S-bit input and querying at most $T < P$ coordinates of its oracle,*

$$\left|\mathsf{P}\left[\mathcal{D}^X(f(X)) = 1\right] - \mathsf{P}\left[\mathcal{D}^{Y_{f(X)}}(f(X)) = 1\right]\right| \le \frac{(S + \log 1/\gamma) \cdot T}{P} + \gamma$$

and

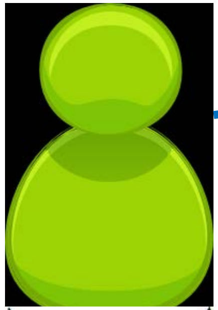
$$\mathsf{P}\left[\mathcal{D}^X(f(X)) = 1\right] \le 2^{(S+2\log 1/\gamma)T/P} \cdot \mathsf{P}\left[\mathcal{D}^{Y_{f(X)}}(f(X)) = 1\right] + 2\gamma .$$

$$\begin{aligned}
\mathsf{P}\left[\mathcal{D}^X(f(X)) = 1\right] &\le \mathsf{P}\left[\mathcal{D}^X(f(X)) = 1, S_{f(X)} \le S + \log 1/\gamma\right] + \mathsf{P}\left[S_{f(X)} > S + \log 1/\gamma\right] \\
&\le \left(2^{(S+2\log 1/\gamma)T/P} \cdot \mathsf{P}\left[\mathcal{D}^{Y_{f(X)}}(f(X)) = 1, S_{f(X)} \le S + \log 1/\gamma\right] + \gamma\right) + \gamma \\
&\le 2^{(S+2\log 1/\gamma)T/P} \cdot \mathsf{P}\left[\mathcal{D}^{Y_{f(X)}}(f(X)) = 1\right] + 2\gamma ,
\end{aligned}$$

Memory Hard Functions, Random Oracles, Graph Pebbling and Extractor Arguments

Jeremiah Blocki

Motivation: Password Storage



jblocki, 123456

Username	Salt	Hash
jblocki	89d978034a3f6	85e23cfe0021f584e3db87aa72630a9a2345c062

SHA1(12345689d978034a3f6)=85e23cfe0021f584e3db87aa72630a9a2345c062

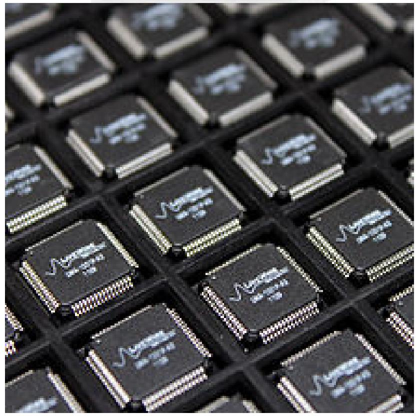
+

Offline Attacks: A Common Problem

- Password breaches at major companies have affected ~~millions~~ **billions** of user accounts.

Goal: Moderately Expensive Hash Function

Fast on PC and
Expensive on ASIC?



PlayStation™

password
hashing
competition

(2013-2015)

https://password-hashing.net/

password
hashing
competition

(2013-2015)

We recommend that you use Argon2…

https://password-hashing.net/

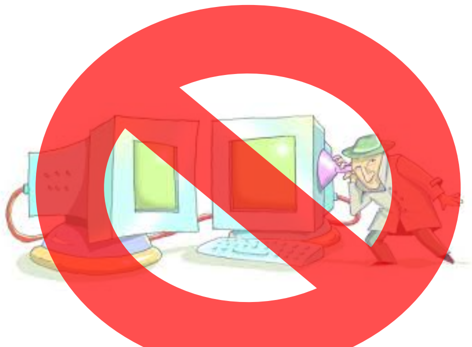
password
hashing
competition

(2013-2015)

https://password-hashing.net/

We recommend that you use Argon2…

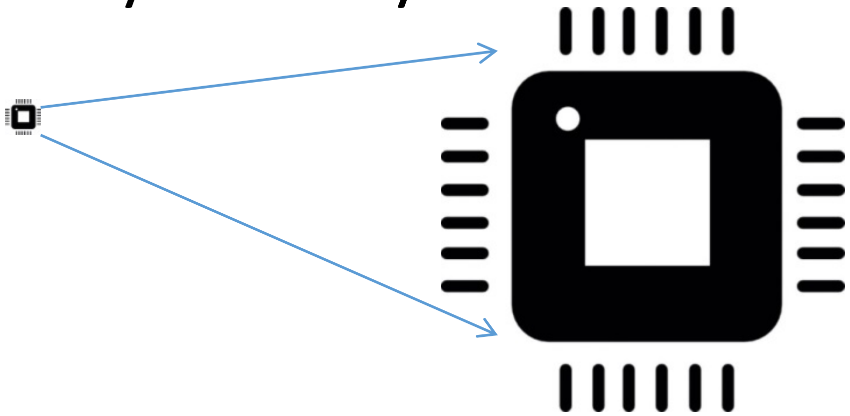
There are two main versions of Argon2, **Argon2i** and Argon2d. **Argon2i** is the safest against side-channel attacks



Memory Hard Function (MHF)

- **Intuition:** computation costs dominated by memory costs

vs.

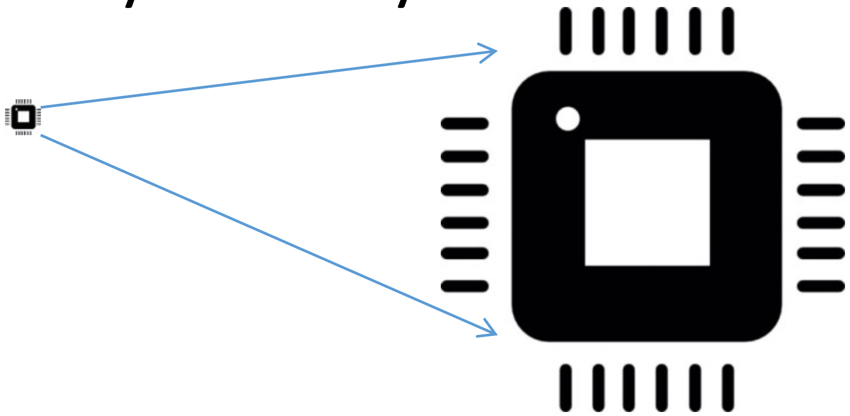


- **Goal:** force attacker to lock up large amounts of memory for duration of computation
  - →Expensive even on customized hardware

Memory Hard Function (MHF)

- **Intuition:** computation costs dominated by memory costs

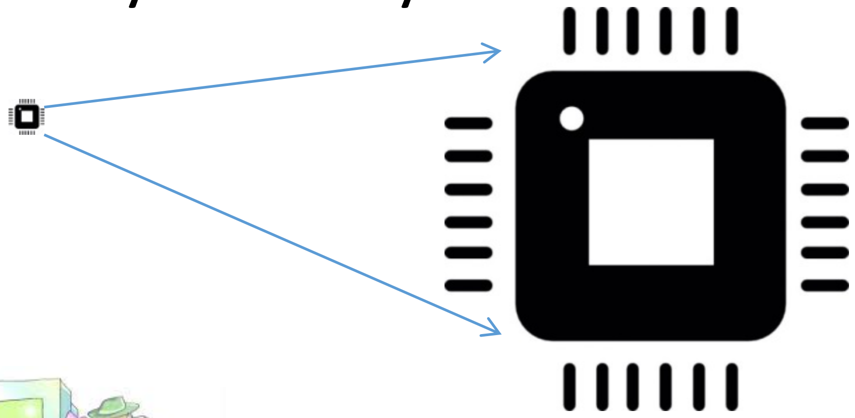
vs.

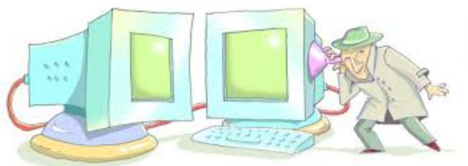


Memory Hard Function (MHF)

- **Intuition:** computation costs dominated by memory costs

vs.

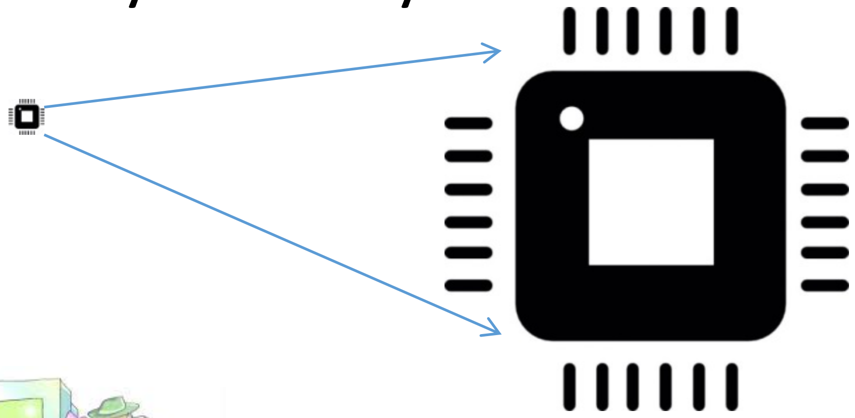




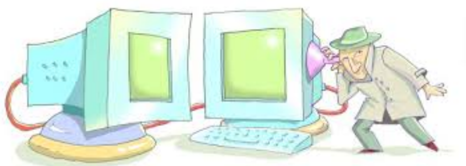
Memory Hard Function (MHF)

- **Intuition:** computation costs dominated by memory costs

vs.



sCrypt

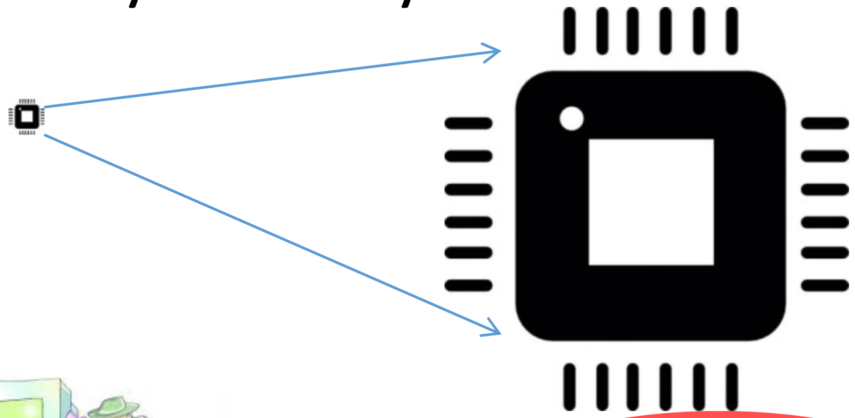


- Data Independent Memory Hard Function (iMHF)
  - Memory access pattern should not depend on input

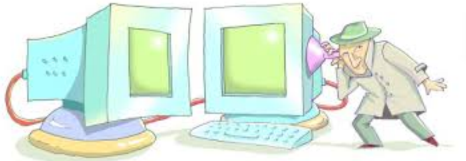
Memory Hard Function (MHF)

- **Intuition:** computation costs dominated by memory costs

vs.

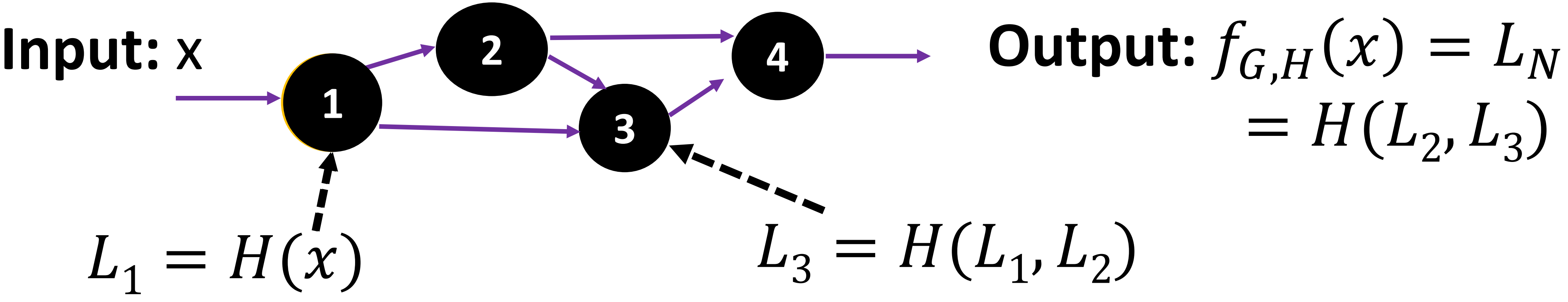


sCrypt



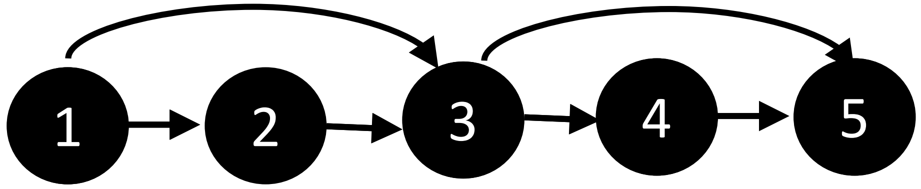
- Data Independent Memory Hard Function (iMHF)
  - Memory access pattern should not depend on input

Data-Independent Memory Hard Function $f_{G,H}$

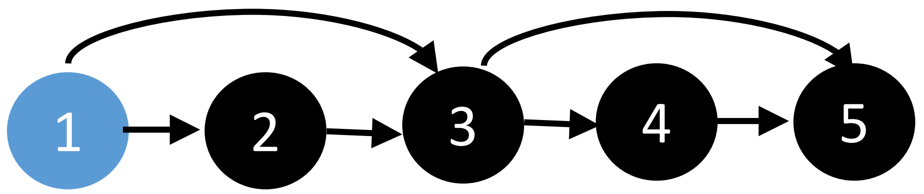


- H:$\{0,1\}^{2k} \rightarrow \{0,1\}^k$ (Random Oracle)
- DAG G (encodes data-dependencies)
  - Maximum indegree: $\delta = O(1)$
  - $\mathrm{N} = 2^{\mathrm{n}}$ nodes

Evaluating an iMHF (pebbling)

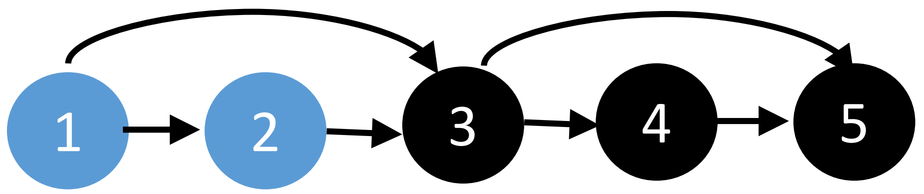


Evaluating an iMHF (pebbling)



$P_1 = \{1\}$ (data value L_1 stored in memory)

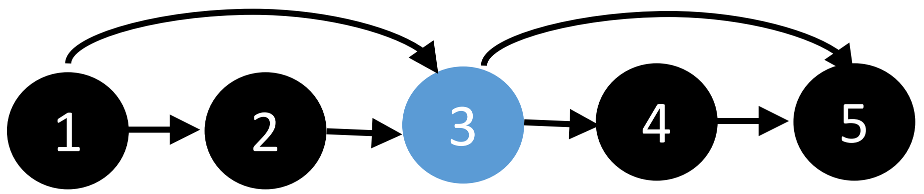
Evaluating an iMHF (pebbling)



$P_1 = \{1\}$

$P_2 = \{1,2\}$ (data values L_1 and L_2 stored in memory)

Evaluating an iMHF (pebbling)

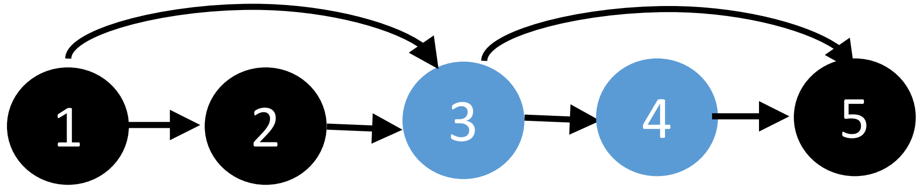


$P_1 = \{1\}$

$P_2 = \{1,2\}$

$P_3 = \{3\}$

Evaluating an iMHF (pebbling)



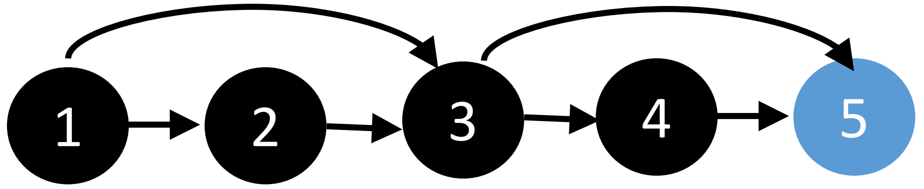
$P_1 = \{1\}$

$P_2 = \{1,2\}$

$P_3 = \{3\}$

$P_4 = \{3,4\}$

Evaluating an iMHF (pebbling)



$P_1 = \{1\}$

$P_2 = \{1,2\}$

$P_3 = \{3\}$

$P_4 = \{3,4\}$

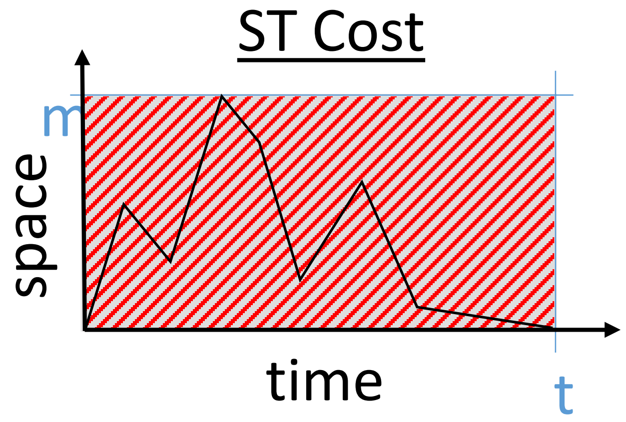
$P_5 = \{5\}$

Measuring Pebbling Cost: Attempt 1

- Space × Time (ST)-Complexity

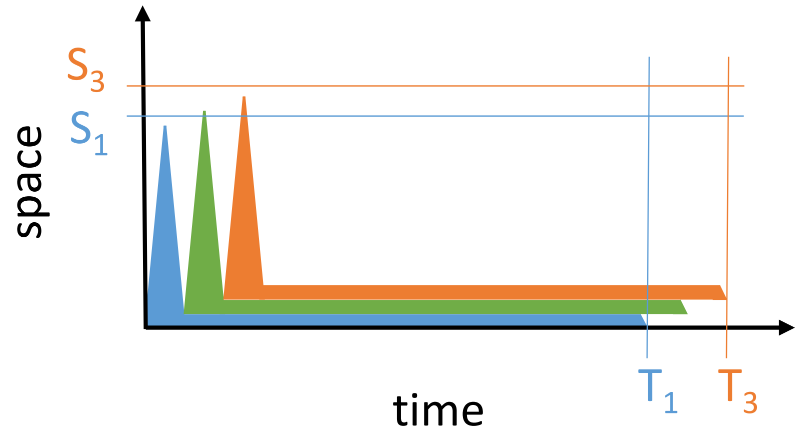
$$\mathrm{ST}(G) = \min_{\vec{P}} \left(t_{\vec{P}} \times \max_{i \le t_{\vec{P}}} |P_i| \right)$$

- Rich Theory
  - Space-time tradeoffs
  - But not appropriate for password hashing



Amortization and Parallelism

- Problem: for parallel computation ST-complexity can scale badly in the number of evaluations of a function.



$$\mathrm{ST}_1 = \mathrm{S}_1 \times \mathrm{T}_1 \approx \mathrm{S}_3 \times \mathrm{T}_3 = \mathrm{ST}_3$$

cost of computing *f* once

cost of computing *f* three times

[AS15] ∃ function f_n (consisting of n RO calls) such that: $ST\left(f^{\times\sqrt{n}}\right) = O(ST(f))$

Measuring Pebbling Costs [AS15]

- Cumulative Complexity (CC)

Memory Used at Step i

Approximates
Amortized Area x Time
Complexity of iMHF

$$\mathrm{CC}(G) = \min_{\vec{P}} \sum_{i=1}^{t_{\vec{P}}} |P_i|$$

- Guessing two passwords doubles the attackers cost

$$\mathrm{CC}(G, G) = 2 \times \mathrm{CC}(G)$$

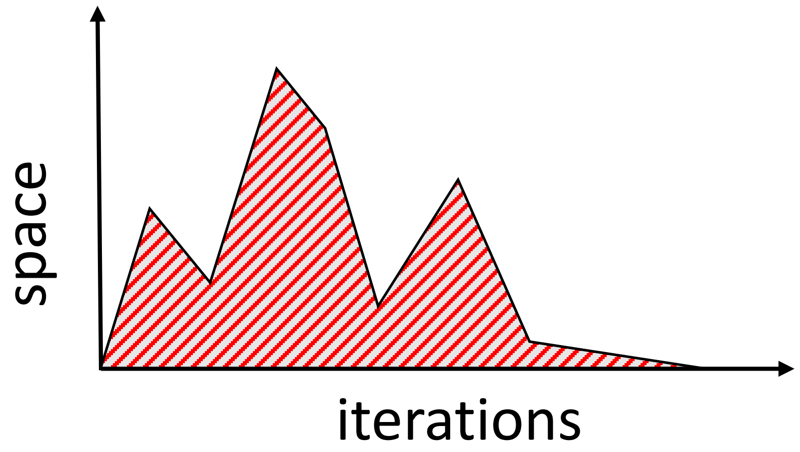
Measuring Pebbling Costs [AS15]

$$\mathrm{CC}(G) = \min_{\vec{P}} \sum_{i=1}^{t_{\vec{P}}} |P_i|$$

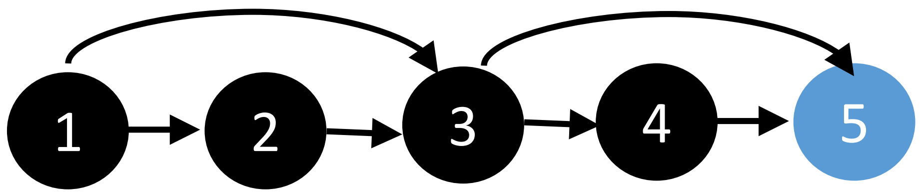
Approximates
Amortized Area x Time
Complexity of iMHF

Memory Used at Step i

Cumulative Memory Cost



Pebbling Example (CC)



$P_1 = \{1\}$

$P_2 = \{1,2\}$

$P_3 = \{3\}$

$P_4 = \{3,4\}$

$P_5 = \{5\}$

$$\mathrm{CC}(G) \leq \sum_{i=1}^{5} |P_i|$$
$$= 1 + 2 + 1 + 2 + 1$$
$$= 7$$

Desiderata

Find a DAG G on n nodes such that

1. Constant Indegree ($\delta = 2$)
   - Running Time: $n(\delta - 1) = n$

2. CC(G) $\geq \frac{n^2}{\tau}$ for some small value τ.

Maximize costs for fixed running time n
(Users are impatient)

DAGs with Maximal CC(G)

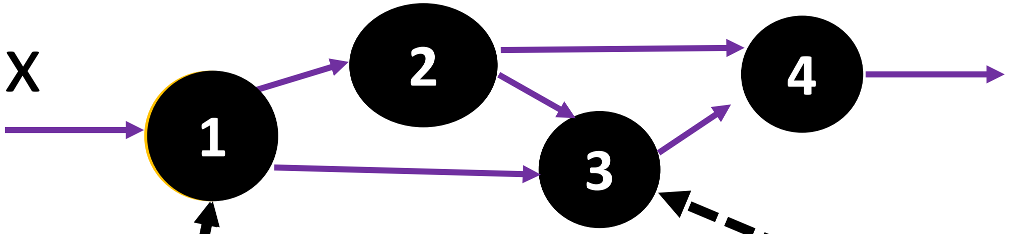
- **Challenge 1:** Design a constant indegree DAG G maximizing CC(G)
  - Depth-Robust Graphs are necessary [AB16] and sufficient [ABP17]
  - Argon2i (PHC winner) is not depth-robust

    → $\mathrm{CC(G)} = \mathrm{o}(n^{1.767}) \ll n^2$ [AB16,AB17,ABP17,BZ17]
  - Any DAG with constant indegree has $\mathrm{CC(G)} = O(n^2 \log\log n \,/\, \log n)$ at most
  - Theoretical [ABP17] then practical [ABH17] construction of depth-robust graphs

    → $\mathrm{CC(G)} = \Omega(n^2/\log n)$ [AB16,AB17,ABP17,BZ17]
- **Open Problem 1:** Construct G with $\mathrm{CC(G)} = \Omega(n^2 \log\log n \,/\, \log n)$
  - Conjecture: [BHKLXZ19] achieves this goal.
- **Open Problem 2:** Tighten constants in upper/lower bounds

Question: CC(G) → cumulative memory cost?

Input: x



Output: $f_{G,H}(x) = L_N$
$= H(L_2, L_3)$

$L_1 = H(x, 0)$

$L_3 = H(L_1, L_2)$

Bad Case: H(x,y)=x+y mod 2^w → $f_{G,H}(x) = k_G \times x$ **e.g.,** $k_G = 3$ **(above)**

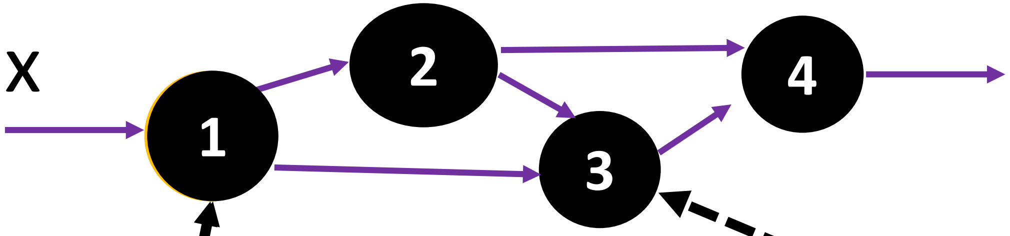
Independent of input! $k_G = 2^{n-2}$ **(complete)**

Computing $f_{G,H}(x)$ is fast + requires minimal memory.

(even if pebbling cost CC(G) is large!)

Question: CC(G) → cumulative memory cost?

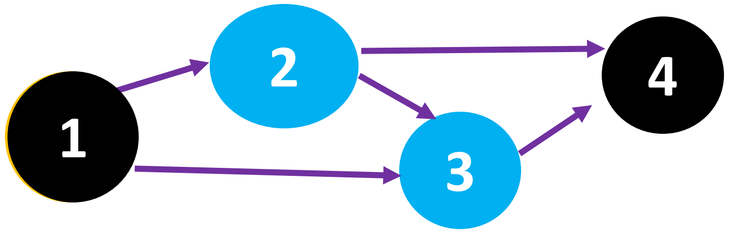
Input: x



Output: $f_{G,H}(x) = L_N$
$= H(L_2, L_3)$

$L_1 = H(x)$

$L_3 = H(L_1, L_2)$



Theorem [AS15]: (in parallel random oracle model)

$A(x) = f_{G,H}(x) \rightarrow \mathrm{cmc}(\mathrm{A}) = \Omega(w \times CC(G))$

Random Oracle Model (PROM)

- Model hash function H as a random function
- Algorithms can only interact with H as an oracle
  - **Query**: x
  - **Response**: H(x)
- If we submit the same query you see the same response
- If x has not been queried, then the value of H(x) is uniform

x	H(x)
$00 \ldots .00$	r_0
$00 \ldots .01$	r_1
…	
…	
$11 \ldots .11$	r_{2^n-1}

- **Real World:** H instantiated as cryptographic hash function (e.g., SHA3) of fixed length (no Merkle-Damgård)

Random Oracle Model: Prediction Game

Prediction Game: $(x_1, y_1), \ldots, (x_k, y_k) \leftarrow A^{H(.)}$ wins the prediction game if

1. $y_1 = H(x_1), \ldots, y_k = H(x_k)$ and
2. the inputs $x_1, \ldots, x_k$ are all fresh i.e., A never queried $H(x_i)$

Fact 1: Any algorithm $A^{H(.)}$ wins the prediction game with probability at most 2^{-kw} over the choice of $H(.)$.

Intuition: A never queries $H(x_1)$ $\rightarrow$ can view $H(x_1)$ as a (yet to be sampled) random string

Random Oracle Model: Prediction Game

Prediction Game: $(x_1, y_1), \ldots, (x_k, y_k) \leftarrow A^{H(.)}$ wins the prediction game if

1. $y_1 = H(x_1), \ldots, y_k = H(x_k)$ and
2. the inputs $x_1, \ldots, x_k$ are all fresh i.e., A never queried $H(x_i)$

Fact 1: Any algorithm $A^{H(.)}$ wins the prediction game with probability at most 2^{-kw} over the choice of $H(.)$.

Fact 2 (Incompressibility of ROs): Any algorithm $A^{H(.)}(h)$ given a s-bit hint h (which may depend on $H(.)$) wins the prediction game with probability at most 2^{-kw+s}

Proof Intuition: Otherwise we can win without hint with probability $> 2^{-kw}$

Reduction: Guess correct hint h with probability 2^{-s} and run $A^{H(.)}(h)$

Parallel Random Oracle Model (PROM)

- PROM Algorithm $\mathcal{A}$(x)
  - **Initial Input/State:** $\sigma_0 = x$
  - $\left(\sigma_1, \overrightarrow{q_1} = \left(x_1^1, \dots, x_{r_1}^1\right)\right) \leftarrow \mathcal{A}(\sigma_0)$
    - New State + Batch of Random Oracle Queries
  - $\overrightarrow{a_1} = \left(H(x_1^1), \dots, H(x_{r_1}^1)\right)$
    - Answers to Random Oracle Queries
  - $\left(\sigma_2, \overrightarrow{q_2} = \left(x_1^2, \dots x_{r_2}^2\right)\right) \leftarrow \mathcal{A}(\sigma_1, \overrightarrow{a_1})$
  - ….
  - $\left(\sigma_i, \overrightarrow{q_i} = \left(x_1^i, \dots x_{r_i}^i\right)\right) \leftarrow \mathcal{A}(\sigma_{i-1}, \overrightarrow{a_{i-1}})$
  - ….
  - $y \leftarrow \mathcal{A}(\sigma_t, \overrightarrow{a_t})$

x	H(x)
$00 \dots .00$	r_0
$00 \dots .01$	r_1
…	
…	
$11 \dots .11$	r_{2^n-1}

One round of computation.

1. $\mathcal{A}$ receives prior answers $\overrightarrow{a_{i-1}}$
2. $\mathcal{A}$ performs arbitrary computation
3. $\mathcal{A}$ outputs $(\sigma_i, \overrightarrow{q_i})$ new state + new queries

Parallel Random Oracle Model (PROM)

- PROM Algorithm $\mathcal{A}$(x)
  - **Fixing $\mathcal{A}$, x and H** we get an execution trace
    $$\mathrm{Trace}_{\mathcal{A},\mathrm{H}}(\mathrm{x}) = \{\sigma_i, \overrightarrow{q_i}, \overrightarrow{a_i}\}_{i=1}^{t}$$
  - Cumulative Memory Cost of Execution Trace
    $$\mathrm{cmc}\left(\mathrm{Trace}_{\mathcal{A},\mathrm{H}}(\mathrm{x})\right) = \sum_{i=1}^{t} (|\sigma_i| + |\overrightarrow{a_i}|)$$
  - Cumulative Memory Cost of a Function
    $$\mathrm{cmc}(f_{G,H}) = \min_{\mathcal{A},\mathrm{x}} \mathbb{E}_H \left[\mathrm{cmc}\left(\mathrm{Trace}_{\mathcal{A},\mathrm{H}}(\mathrm{x})\right)\right]$$

x	H(x)
00 … .00	r_0
00 … .01	r_1
…	
…	
11 … .11	r_{2^n-1}

Min over inputs x and PROM algorithms $\mathcal{A}$ evaluating $f_{G,H}$

Expectation over selection of random oracle

Collision Problem

Collision Problem: Suppose that we are asked to find $x \neq x'$ s.t. $H(x) = H(x')$

x	H(x)
$00 \ldots .00$	r_0
$00 \ldots .01$	r_1
…	
…	
$11 \ldots .11$	$r_{2^n - 1}$

What is the probability we can succeed given q queries to the random oracle?

Answer: $\leq q^2 2^{-w}$

Explanation: Let $\boldsymbol{x_1}, \ldots, \boldsymbol{x_q}$ *be the queries we make*

$$\mathbf{Pr}[H(x_{i+1}) \in \{H(x_1), \ldots, H(x_i)\}] \leq i \times 2^{-w}$$ (Prob Collision at time i+1)

$$\therefore \ \mathbf{Pr}[Collision] \leq \sum_{\boldsymbol{i \leq q}} i \times 2^{-w}$$ (Union Bound over Each Round)

Label Distinctness

Label Distinctness: Suppose we are given a directed acyclic graph G on n nodes V={1,…,n} with indegree 2 and such that each node v > 2 has two parents v-1 and r(v)<v-1. Let

x	H(x)
$00\ldots.00$	r_0
$00\ldots.01$	r_1
…	
…	
$11\ldots.11$	r_{2^n-1}

Let $x = L_0$ be the initial input (w-bits) and define labels $L_1 = H(x_0, 0^w), L_2 = H(L_1, 0^w)$,

$$L_3 = H(L_2, L_1)$$
$$\ldots$$
$$L_v = H\left(L_{v-1}, L_{r(v)}\right)$$
$$\ldots$$
$$L_n = H\left(L_{n-1}, L_{r(n)}\right)$$

Question: What is the probability that two labels collide?

Label Distinctness

$$\ldots$$
$$L_v = H\left(L_{v-1}, L_{r(v)}\right)$$
$$\ldots$$

Question: What is the probability that two labels collide?

Let $\mathbf{U}_{\boldsymbol{i}}$ be the event that labels $L_1, \ldots, L_i$ are all distinct

$$\Pr[\overline{\mathbf{U}_{\boldsymbol{i}}} | \mathbf{U}_{\boldsymbol{i-1}}] = \Pr\left[H\left(L_{i-1}, L_{r(i)}\right) \in \{L_1, \ldots, L_{i-1}\} | \mathbf{U}_{\boldsymbol{i-1}}\right] \leq (i-1)2^{-w}$$

L_{i-1} unique → fresh query!

$Union\ Bound!$

Label Distinctness

$$\ldots$$
$$L_v = H\big(L_{v-1}, L_{r(v)}\big)$$
$$\ldots$$

Question: What is the probability that two labels collide?

Let $\mathbf{U}_{\boldsymbol{i}}$ be the event that labels $L_1, \ldots, L_i$ are all distinct

$$\Pr[\overline{\mathbf{U}_{\boldsymbol{i}}} | \mathbf{U}_{\boldsymbol{i-1}}] \leq (i-1)2^{-w}$$

$$\Pr[\overline{\mathbf{U}_{\boldsymbol{n}}}] \leq \sum_{i \leq n} (i-1)\, 2^{-w} \leq n^2 2^{-w}$$

Label Collision

$$\dots$$

$$L_v = H(\underbrace{L_{v-1}, L_{r(v)}}_{})$$

$$\mathbf{Prelab}(\boldsymbol{v}) = L_{v-1}, L_{r(v)}$$

x	H(x)
$00 \dots .00$	r_0
$00 \dots .01$	r_1
…	
…	
$11 \dots .11$	r_{2^n-1}

Question: Suppose we can make at most q queries to the random oracle. What is the probability we find some z s.t. $L_v = H(z)$ but $\mathrm{z} \neq \mathbf{Prelab}(\mathbf{v})$ for some node v?

Label Collision

Question: Suppose we can make at most q queries to the random oracle. What is the probability we find some z s.t. $L_v = H(z)$ but $\mathrm{z} \neq \mathbf{Prelab}(\mathbf{v})$ for some node v?

Answer: at most $nq2^{-w}$

Let z_i be ith query to random oracle such that $z_i \neq \mathbf{Prelab}(\boldsymbol{v})$ for any node $\boldsymbol{v} \leq \boldsymbol{n}$ then we have

$$\mathbf{Pr}[H(z_i) \in \{L_1, \dots, L_n\}] \leq n2^{-w}$$

$$\mathbf{Pr}[\exists \boldsymbol{i} \leq q. H(z_i) \in \{L_1, \dots, L_n\}] \leq nq2^{-w}$$

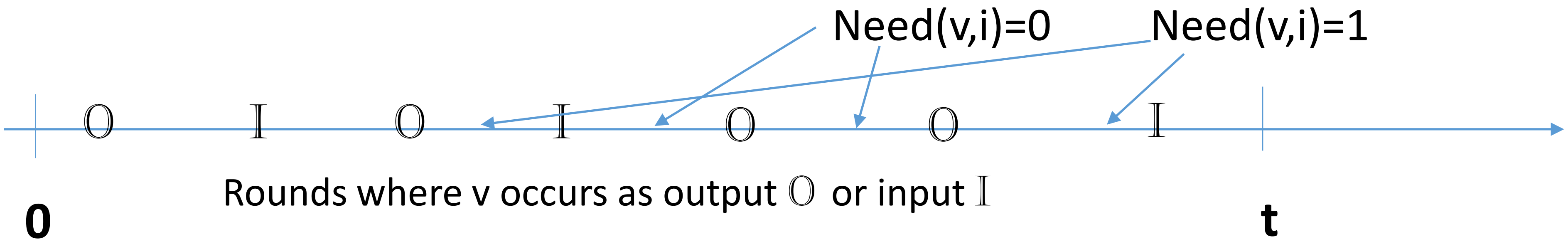
x	H(x)
$00 \dots .00$	r_0
$00 \dots .01$	r_1
…	
…	
$11 \dots .11$	r_{2^n-1}

Ex Post Facto Pebbling

- **Fixing $\mathcal{A}$, x and H** we get an execution trace

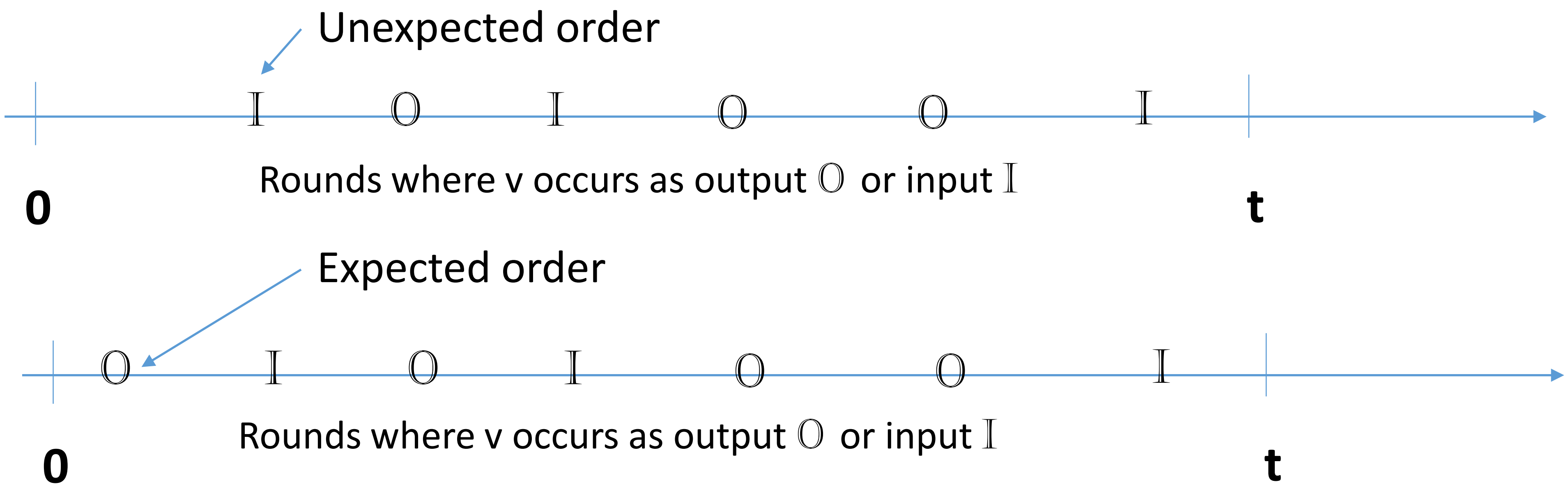
$$\text{Trace}_{\mathcal{A},\mathrm{H}}(\mathrm{x}) = \{\sigma_i, \overrightarrow{q_i}, \overrightarrow{a_i}\}_{i=1}^{t}$$

- Track L_v for each node v
  - Note rounds where L_v appear as the input to random oracle query?
  - Note rounds does L_v appear as an the output to a random oracle query?
  - Define Need(v,i)=1 if and only if the next time (after round i) label L_v appears it is as an input; otherwise Need(v,i)=0
- $\mathrm{P_i} = \{\mathrm{v} : \mathrm{Need}(\mathrm{v}, \mathrm{i}) = 1\}$



Ex Post Facto Pebbling

- $\mathrm{P_i} = \{\mathrm{v} : \mathrm{Need}(\mathrm{v}, \mathrm{i}) = 1\}$ → does this give us a legal pebbling?
- Order(v) be the bad event L_v is used as an RO input before it has appeared as an output



Ex Post Facto Pebbling

- $\mathrm{P_i} = \{\mathrm{v} : \mathrm{Need}(\mathrm{v}, \mathrm{i}) = 1\}$ → does this give us a legal pebbling?

Claim 1: Suppose that $\mathcal{A}$ computes $f_{G,H}$ and makes at most q random oracle queries then $\mathrm{P} = P_1, \ldots, P_t$ is a legal pebbling (except with probability $\mathrm{O}(qn2^{-w})$.

Proof Sketch:

Observation 1: If the bad event Order(v) never occurs for any node v then the pebbling is legal (follows from definition of Need(v,i))

Ex Post Facto Pebbling

- $\mathrm{P_i} = \{\mathrm{v} : \mathrm{Need}(\mathrm{v}, \mathrm{i}) = 1\}$ → does this give us a legal pebbling?

Claim 1: Suppose that $\mathcal{A}$ computes $f_{G,H}$ and makes at most q random oracle queries then $\mathrm{P} = P_1, \ldots, P_t$ is a legal pebbling (except with probability $\mathrm{O}(qn2^{-w})$.

Proof Sketch:

Observation 2: If L_v has not yet appeared as output then the probability a particular query includes L_v as input early is at most 2^{-w}

→$\mathrm{Pr}[\mathrm{Order}(\mathrm{v})] \leq q2^{-w}$ (Union Bound over all q queries)

(L_v can be viewed as random w-bit string before it first appears)

Ex Post Facto Pebbling

- $P_i = \{v : \mathrm{Need}(v, i) = 1\}$ → does this give us a legal pebbling?

Claim 1: Suppose that $\mathcal{A}$ computes $f_{G,H}$ and makes at most q random oracle queries then $\mathrm{P} = P_1, \dots, P_t$ is a legal pebbling (except with probability $\mathrm{O}(qn2^{-w})$.

Proof Sketch: Let Order(v) be the bad event L_v is used as an RO input before it has appeared as an output. Union Bounding

$$\Pr[\exists v\ Order(v)\ ] \leq \mathrm{n}\Pr[Order(v)] \leq nq2^{-w}$$

(L_v can be viewed as random w-bit string before it first appears)

Ex Post Facto Pebbling

- **Fixing $\mathcal{A}$, x and H** we get an execution trace

$$\mathrm{Trace}_{\mathcal{A},\mathrm{H}}(\mathrm{x}) = \{\sigma_i, \overrightarrow{q_i}, \overrightarrow{a_i}\}_{i=1}^{t}$$

Claim 1: Suppose that $\mathcal{A}$ computes $f_{G,H}$ and makes at most q random oracle queries then $\mathrm{P} = P_1, \ldots, P_t$ is a legal pebbling (except with probability $\mathrm{O}(qn2^{-w})$.

Observation: If P is legal then $\mathrm{CC}(\mathrm{P}) \geq CC(G)$

(definition of $CC(G)$ as best pebbling of G)

Extractor Argument

- **Fixing $\mathcal{A}$, x and H** we get an execution trace

$$\mathrm{Trace}_{\mathcal{A},\mathrm{H}}(\mathrm{x}) = \{\sigma_i, \overrightarrow{q_i}, \overrightarrow{a_i}\}_{i=1}^{t}$$

Observation: $\mathrm{CC}(\mathrm{P}) \geq CC(G)$ (definition of $CC(G)$)

Claim 2: For each round i we have $|\sigma_i| + |\overrightarrow{a_{i-1}}| \geq \mathrm{w}|P_i|/2$

Proof Idea: Extractor argument. Suppose for contradiction that $|\sigma_i| + |\overrightarrow{a_{i-1}}| < \mathrm{w}|P_i|/2$.

We will build an extractor that outputs $|P_i|$ labels given a hint of size $\mathrm{w}|P_i|/2 + o(\mathrm{w}|P_i|)$. This yields a contradiction of incompressibility!

Extractor Hint

Claim 2: For each round i we have $|\sigma_i| + |a_{i-1}| \geq \mathrm{w}|P_i|/2$

Hint: h

- Initial State: $\sigma_i, \overrightarrow{a_{i-1}}$ (used to simulate $\mathcal{A}$ at most $\mathrm{w}|P_i|/2$ bits)
- Encoding of P_i ($|P_i| \log n$ bits)
- For each $\mathrm{v} \in P_i$ index i_v of next random oracle query where label L_v appears as input ($|P_i| \log q$ bits)
- For each $\mathrm{v} \in P_i$ index o_v of next random oracle query where label L_v appears as output ($|P_i| \log q$ bits)
- **Total Hint Length:** $\mathrm{w}|P_i|/2 + o(\mathrm{w}|P_t|)$.

Extractor argument. Suppose for contradiction that $|\sigma_i| < \mathrm{w}|P_t|/2$.

We will build an extractor that outputs $|P_t|$ labels given a hint of size $|\sigma_i| + o(\mathrm{w}|P_t|)$. This yields a contradiction of incompressibility!

Extractor: Simulating Attacker

Random Oracle
$H: \{0,1\}^* \rightarrow \{0,1\}^w$

P_i	Label	Input Query	Output Query
v_1	???	(i+2,4)	(i+10,5)
v_2	???	(i+1,2)	(i+2,1)
…	…		

Extractor E
Hint: h = $(\sigma_i, \overrightarrow{a_{i-1}}\ P_i, \ldots)$

$\sigma_{i+1}, \overrightarrow{q_i},$

Simulate
$\mathcal{A}(\sigma_i, \overrightarrow{a_{i-1}})$

$\overrightarrow{a_i} = (H(\overrightarrow{q_i}[1]), H(\overrightarrow{q_i}[2]), \ldots,)$

Extractor: Simulating Attacker

Random Oracle
$H: \{0,1\}^* \rightarrow \{0,1\}^w$

Extractor E
Hint: h = $(\sigma_i, \overrightarrow{a_{i-1}}\ P_i, \ldots)$

$\sigma_{i+2}, \overrightarrow{\boldsymbol{q_{i+1}}},$

Simulate
$\mathcal{A}(\sigma_{i+1}, \overrightarrow{a_i})$

P_i	Label	Input Query	Output Query
v_1	???	(i+2,4)	(i+10,5)
v_2	???	**(i+1,2)**	(i+2,1)
…	…		

Extract!
$\overrightarrow{q_{i+1}}$[2] contains $\boldsymbol{L_{v_2}}$

Extractor: Simulating Attacker

Random Oracle
$H: \{0,1\}^* \rightarrow \{0,1\}^w$

P_i	Label	Input Query	Output Query
v_1	???	(i+2,4)	(i+10,5)
v_2	$\boldsymbol{L_{v_2}}$	(i+1,2)	(i+2,1)
…	…		

Extractor E
Hint: h = $(\sigma_i, \overrightarrow{a_{i-1}}\ P_i, \ldots)$

$\sigma_{i+2}, \overrightarrow{q_{i+1}},$

$\overrightarrow{a_{i+1}} = (H(\overrightarrow{q_{i+1}}[1]), H(\overrightarrow{q_{i+1}}[2]), \ldots,)$

Simulate
$\mathcal{A}(\sigma_{i+1}, \overrightarrow{a_i})$

Extract!
$\overrightarrow{q_{i+1}}[2]$ contains $\boldsymbol{L_{v_2}}$

Extractor: Simulating Attacker

Random Oracle
$H: \{0,1\}^* \rightarrow \{0,1\}^w$

Extractor E
Hint: h = (σ_i, $\overrightarrow{a_{i-1}}$ P_i, …)

$\sigma_{i+3}, \overrightarrow{q_{i+2}}$

Simulate
$\mathcal{A}(\sigma_{i+2}, \overrightarrow{a_{i+1}})$

P_i	Label	Input Query	Output Query
v_1	???	**(i+2,4)**	(i+10,5)
v_2	$\boldsymbol{L_{v_2}}$	(i+1,2)	**(i+2,1)**
…	…		

Extract! $\mathrm{H}(\overrightarrow{q_{i+2}}[4]) = \boldsymbol{L_{v_1}}$

Extractor: Simulating Attacker

Random Oracle
$H: \{0,1\}^* \rightarrow \{0,1\}^w$

Extractor E
Hint: h = (σ_i, $\overrightarrow{a_{i-1}}$ P_i, …)

$\sigma_{i+3}, \overrightarrow{q_{i+2}}$

$\overrightarrow{a_{i+2}} = (H(\overrightarrow{q_{i+2}}[1]), H(\overrightarrow{q_{i+2}}[2]), \dots,)$

Simulate
$\mathcal{A}(\sigma_{i+2}, \overrightarrow{a_{i+1}})$

P_i	Label	Input Query	Output Query
v_1	$\boldsymbol{L_{v_1}}$	(i+2,4)	(i+10,5)
v_2	$\boldsymbol{L_{v_2}}$	(i+1,2)	**(i+2,1)**
…	…		

Danger! $\mathrm{H}(\overrightarrow{q_{i+2}}[1]) = \boldsymbol{L_{v_2}}$
Do not submit this query!

Extractor: Simulating Attacker

Random Oracle
$H: \{0,1\}^* \rightarrow \{0,1\}^w$

Extractor E
Hint: h = (σ_i, $\overrightarrow{a_{i-1}}$ P_i, ...)

$\sigma_{i+3}, \overrightarrow{q_{i+2}}$

Simulate
$\mathcal{A}(\sigma_{i+2}, \overrightarrow{a_{i+1}})$

$\overrightarrow{a_{i+2}} = \left(\boldsymbol{L_{v_2}}, H(\overrightarrow{q_{i+2}}[2]), \dots, \right)$

P_i	Label	Input Query	Output of Query
v_1	$\boldsymbol{L_{v_1}}$	(i+2,4)	(i+10,5)
v_2	$\boldsymbol{L_{v_2}}$	(i+1,2)	$\overrightarrow{q_{i+2}}[1]$
…	…		

Danger! $\mathrm{H}(\overrightarrow{q_{i+2}}[1]) = \boldsymbol{L_{v_2}}$
Do not submit this query!

Extractor: Simulating Attacker

Random Oracle
$H: \{0,1\}^* \to \{0,1\}^w$

Extractor E
Hint: h = (σ_i, $\overrightarrow{a_{i-1}}$ P_i, …)

$\sigma_{i+4}, \overrightarrow{q_{i+3}}$

Simulate
$\mathcal{A}(\sigma_{i+3}, \overrightarrow{a_{i+2}})$

$\overrightarrow{a_{i+3}} = (H(\overrightarrow{q_{i+3}}[1]), H(\overrightarrow{q_{i+3}}[2]), \dots,)$

P_i	Label	Input Query	Output of Query
v_1	$\boldsymbol{L_{v_1}}$	(i+2,4)	(i+10,5)
v_2	$\boldsymbol{L_{v_2}}$	(i+1,2)	$\overrightarrow{q_{i+2}}[1]$
…	…		

etc …

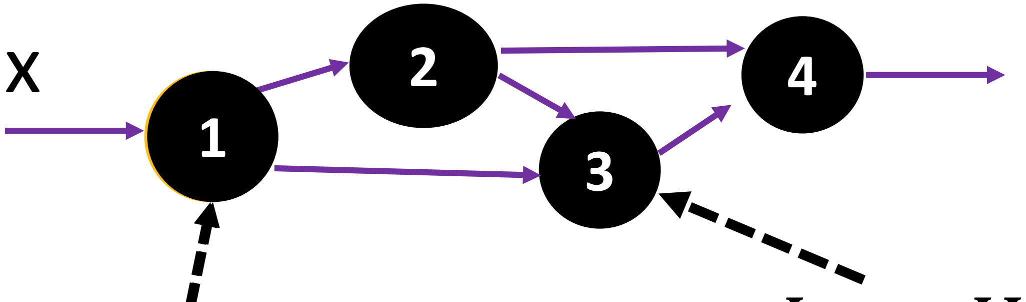
Extractor: Simulating Attacker

Claim 2: For each round i we have $|\sigma_i| \geq \mathrm{w}|P_i|/2$

Hint:

- simulate $\mathcal{A}$ from initial state: σ_i
  - Forward random oracle queries to $\mathrm{H}(.)$ (* One Exception Below *)
- For each $\mathrm{v} \in P_i$ wait for first query where L_v appears as input and record L_v (by definition of P_i this occurs before L_v appears as output)
- For each $\mathrm{v} \in P_i$ wait for first query o_v which produces output L_v
  - Do not forward this query to $\mathrm{H}(.)$
  - Simply record the response L_v
  - Technical Note: Extractor can simply run naïve evaluation algorithm for $f_{G,H}(x)$ after simulating $\mathcal{A}$ to ensure that for each $\mathrm{v} \in P_i$ there is some round where L_v is output

Input: x



Output: $f_{G,H}(x) = L_N$
$= H(L_2, L_3)$

$L_1 = H(x)$

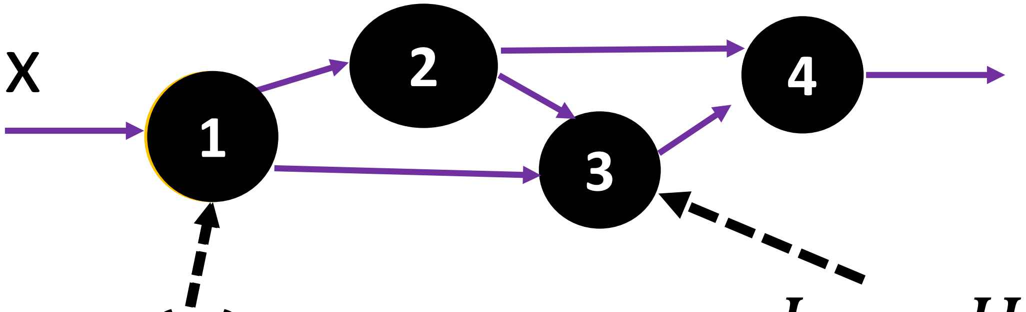
$L_3 = H(L_1, L_2)$

Extractor:

- Outputs L_v for each v $\in P_i$
- Generate remaining labels L_v for each v $\notin P_i$
  - Can be done querying random oracle at x_v s.t. $H(x_v) = L_v$
- Yields k ``fresh" input output pairs (x_v, L_v) for each v $\in P_i$ as long as all labels L_v are distinct

$$\Pr[\exists(u, v).\ L_v = L_u] \leq n^2 2^{-w}$$

Input: x



Output: $f_{G,H}(x) = L_N$
$= H(L_2, L_3)$

$L_1 = H(x)$

$L_3 = H(L_1, L_2)$

Extractor: Yields $\mathrm{k} = |P_i|$ ``fresh" input output pairs (x_v, L_v) for each $\mathrm{v} \in P_i$ as long as all labels L_v are distinct and pebbling is legal

$$\Pr[\exists (u, v).\ L_v = L_u] \le n^2 2^{-w}$$

$$\rightarrow \Pr[Success] \ge 1 - n^2 2^{-w} - qn2^{-w}$$

Contradiction! Extractor can succeed with probability at most $2^{-kw/2+o(kw)}$

Reflection: Extractor Argument

- What properties of the random oracle did we use?

- Simulatability/Delayed Sampling:
  - Can view H(x) as uniformly random string that is yet to be sampled
    - (until x is actually queried)
    - used to analyze the probability that a label L_v appears out of order (also collisions)
- Extractability of Queries:
  - When attacker submits random oracle query the extractor gets to see the query (and the response)

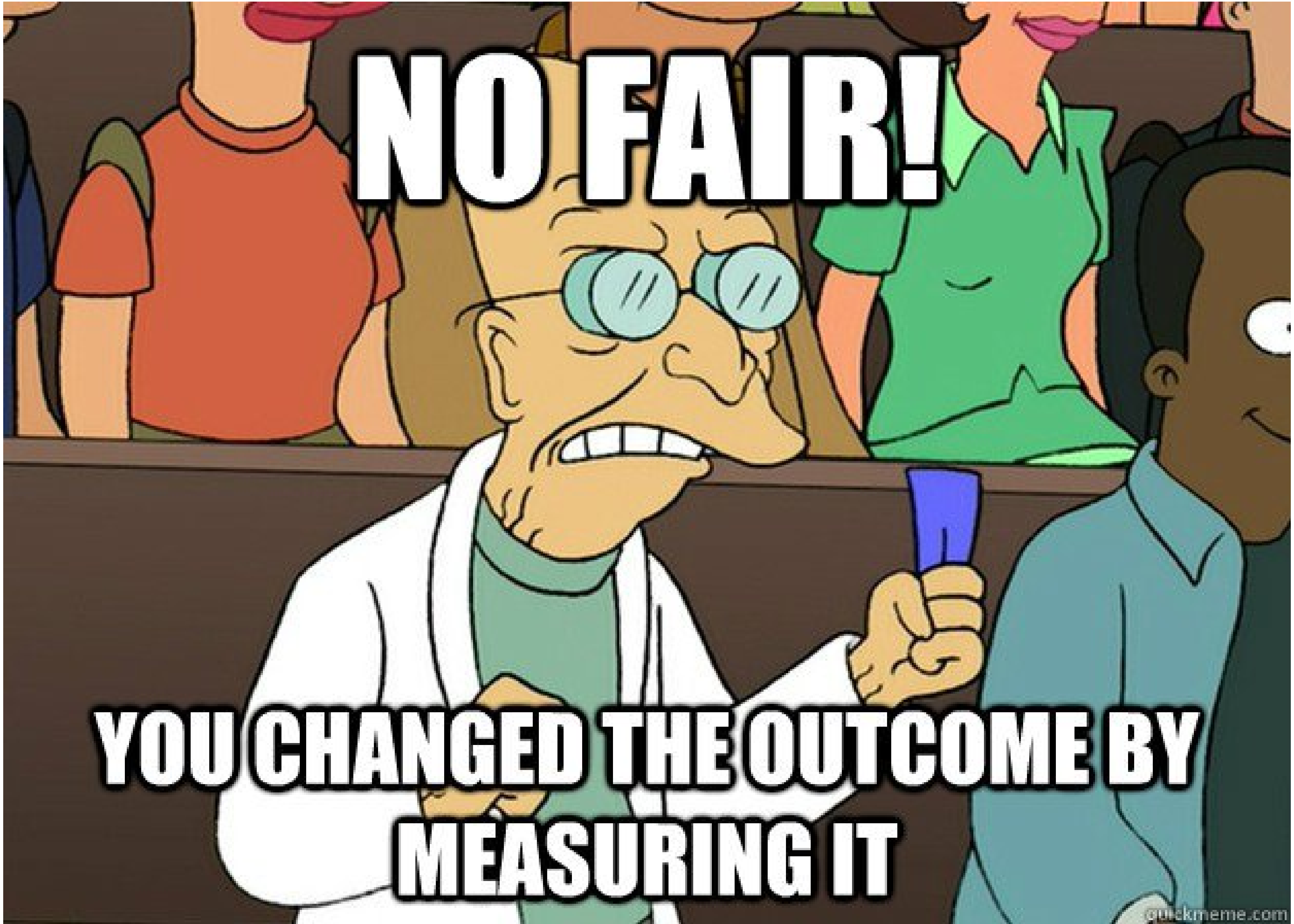
Quantum Random Oracle Model

- *Similar to classical random oracle model except that input is an entangled quantum state*

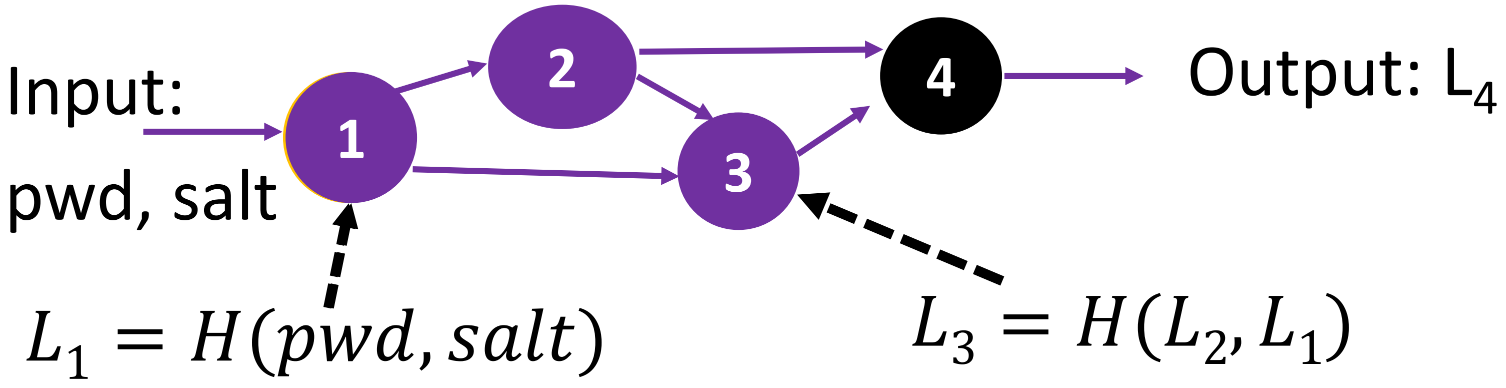
$$\sum_x \alpha_x |x, y\rangle \underset{H}{\rightarrow} \sum_x \alpha_x |x, y \oplus H(x)\rangle$$

- Realistic model for <u>any</u> realization of the random oracle e.g., can implement SHA3 as a quantum circuit

- **Challenge:** extractor needs to view random oracle queries


NO FAIR!
YOU CHANGED THE OUTCOME BY MEASURING IT
quickmeme.com

Evaluating an iMHF (pebbling)



Pebbling Rules : $\vec{P}=P_1,\ldots,P_t \subset V$ s.t.

- $P_{i+1} \subset P_i \cup \{x \in V \mid \mathrm{parents}(x) \subset P_{i+1}\}$ (need dependent values)
- $n \in P_t$ (must finish and output L_n)

Measuring Pebbling Costs [AS15]

- Cumulative Complexity (CC)

Memory Used at Step i

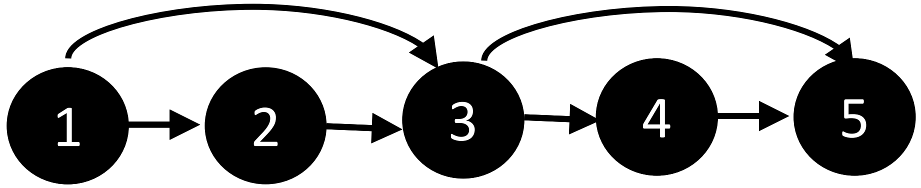
Approximates
Amortized Area x Time
Complexity of iMHF

$$CC(G) = \min_{\vec{P}} \sum_{i=1}^{t_{\vec{P}}} |P_i|$$

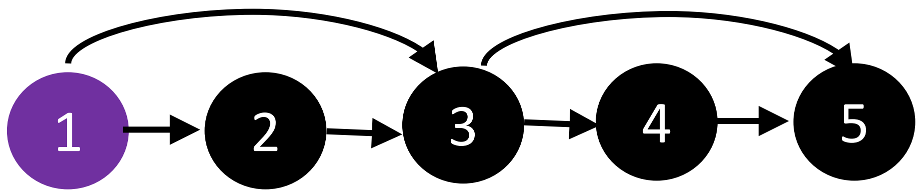
- Guessing two passwords doubles the attackers cost

$$CC(G, G) = 2 \times CC(G)$$

Naïve: Pebbling Strategy

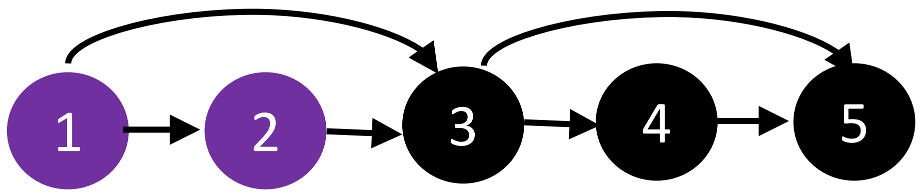


Naïve: Pebbling Strategy



$P_1 = \{1\}$

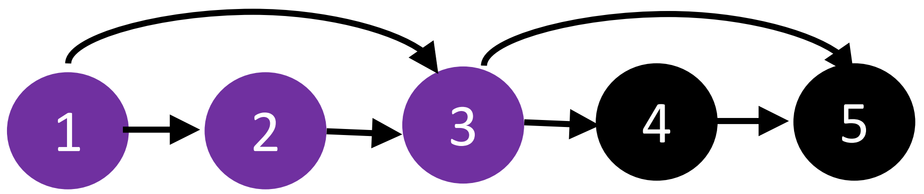
Naïve: Pebbling Strategy



$P_1 = \{1\}$

$P_2 = \{1,2\}$

Naïve: Pebbling Strategy

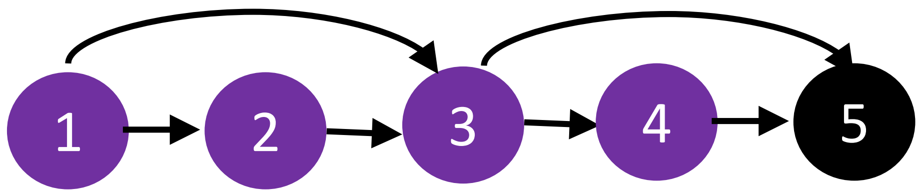


$P_1 = \{1\}$

$P_2 = \{1,2\}$

$P_3 = \{1,2, 3\}$

Naïve: Pebbling Strategy



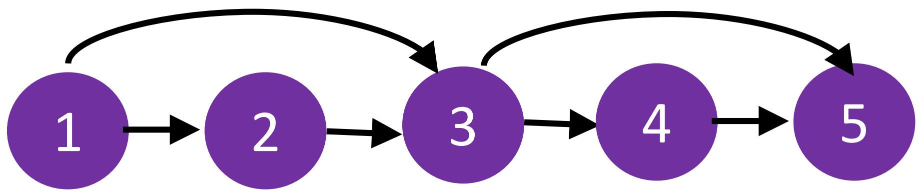
$P_1 = \{1\}$

$P_2 = \{1,2\}$

$P_3 = \{1,2, 3\}$

$P_4 = \{1, 2, 3, 4\}$

Naïve: Pebbling Strategy



$P_1 = \{1\}$

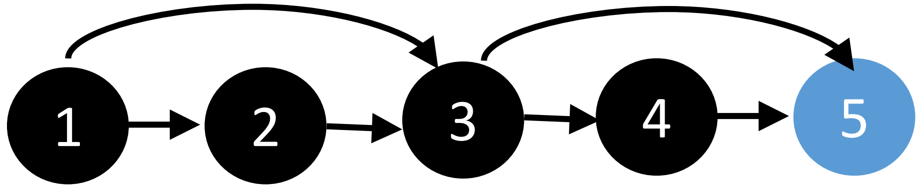
$P_2 = \{1,2\}$

$P_3 = \{1,2, 3\}$

$P_4 = \{1, 2, 3, 4\}$

$P_5 = \{1, 2, 3, 4, 5\}$

Naïve: Pebbling Strategy (CC)



$P_1 = \{1\}$

$P_2 = \{1,2\}$

$P_3 = \{1,2, 3\}$

$P_4 = \{1,2, 3,4\}$

$P_5 = \{1,2,3,4,5\}$

$$CC(G) \leq \sum_{i=1}^{5} |P_i|$$

$$= 1 + 2 + 3 + 4 + 5$$

$$= 15$$

Naïve Pebbling Algorithms

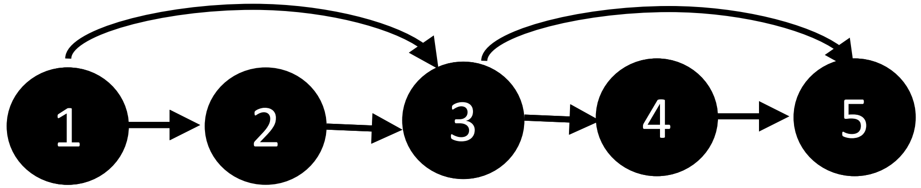
- Naïve (Pebble in Topological Order)
  - Never discard pebbles
  - Legal Pebbling Strategy for any DAG!
  - Pebbling Time: n
  - Sequential: Place one new pebble on the graph in each round

Theorem: *Any DAG G has* $CC(G) \leq \sum_{i \leq n} i = \frac{n(n+1)}{2}$

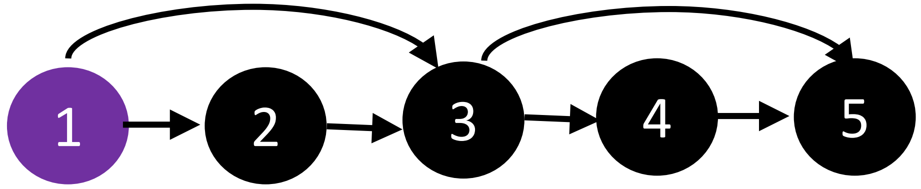
Proof: *Naïve pebbling strategy is legal strategy for any DAG G!*

Question: Can we find a DAG G with $CC(G) = \Omega(n^2)$?

Improved Pebbling

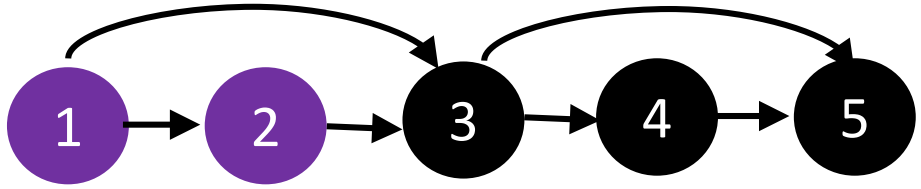


Improved Pebbling



$P_1 = \{1\}$

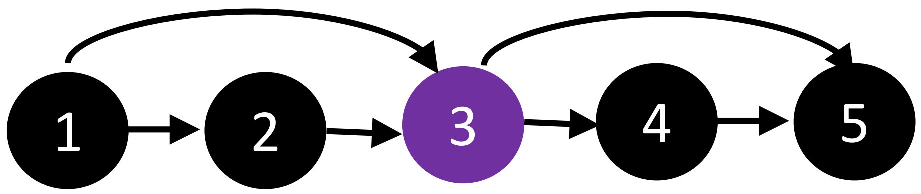
Improved Pebbling



$P_1 = \{1\}$

$P_2 = \{1,2\}$

Improved Pebbling

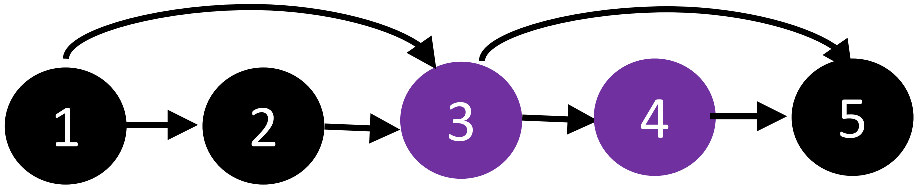


$P_1 = \{1\}$

$P_2 = \{1,2\}$

$P_3 = \{3\}$

Improved Pebbling



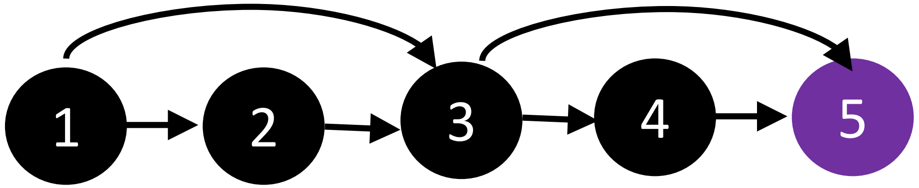
$P_1 = \{1\}$

$P_2 = \{1,2\}$

$P_3 = \{3\}$

$P_4 = \{3,4\}$

Improved Pebbling



$P_1 = \{1\}$

$P_2 = \{1,2\}$

$P_3 = \{3\}$

$P_4 = \{3,4\}$

$P_5 = \{5\}$

Graphs with High CC

Theorem: *Any DAG G has* $CC(G) \leq \sum_{i \leq n} i = \frac{n(n+1)}{2}$
Proof: *Naïve pebbling strategy is legal strategy for any DAG G!*

Question: Can we find a DAG G with $CC(G) = \Omega(n^2)$?

Claim: The complete DAG has $CC(G) \geq \sum_{i \leq n-1} i = \frac{n(n-1)}{2} = \Omega(n^2)$?

Proof: Consider the round immediately before we first place a pebble on node i+1. We must have had pebbles on all of the nodes {1,…,i}.

Question: Can we find a DAG G with $CC(G) = \Omega(n^2)$ and low indegree?

Why do we care about indegree?

In practice the random oracle is instantiated with a function $\boldsymbol{H}: \{\mathbf{0}, \mathbf{1}\}^{2\lambda} \rightarrow \{\mathbf{0}, \mathbf{1}\}^{\lambda}$
Label of node v is obtained by hashing labels of v's parents.

Node v has two parents (u and w) ➔ $\boldsymbol{L_v} = \boldsymbol{H}(\boldsymbol{L_u}, \boldsymbol{L_w})$ ➔ One oracle to H used to compute label

Node v has three parents (u, w, x) ➔ $\boldsymbol{L_v} = \boldsymbol{H}(\boldsymbol{H}(\boldsymbol{L_u}, \boldsymbol{L_w}), \boldsymbol{L_x})$ ➔ Two oracle queries to H to compute label

Node v has four parents (u, w, x, y) ➔ $\boldsymbol{L_v} = \boldsymbol{H}(\boldsymbol{H}(\boldsymbol{H}(\boldsymbol{L_u}, \boldsymbol{L_w}), \boldsymbol{L_x}), \boldsymbol{L_y})$ ➔ Three oracle queries to H to compute label

Node v has k parents ➔ k-1 oracle queries to H to compute label

Running time to evaluate $\boldsymbol{f_{G,H}}$ is proportional to $\boldsymbol{n} \times \boldsymbol{indeg}(\boldsymbol{G})$

Desiderata

Find a DAG G on n nodes such that

1. Constant Indegree ($\delta = 2$)
   - Running Time: $n(\delta - 1) = n$

2. CC(G) $\geq \frac{n^2}{\tau}$ for some small value τ.

Maximize costs for fixed running time n
(Users are impatient)

Outline

- Motivation
- Data Independent Memory Hard Functions (iMHFs)
- **Our Attacks**
  - General Attack on Non Depth Robust DAGs
  - Existing iMHFs are not Depth Robust
  - Ideal iMHFs don’t exist
- Subsequent Results (Depth-Robustness is Sufficient)
- Open Questions

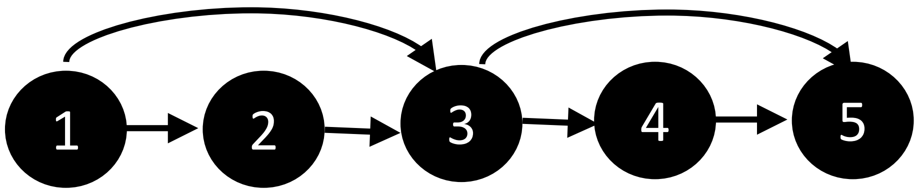
Depth-Robustness: A Necessary Property

Depth Robustness

Definition: A DAG G=(V,E) is (e,d)-reducible if there exists $S \subseteq V$ s.t. $|S| \leq e$ and depth(G-S) $\leq$ d.

Otherwise, we say that G is (e,d)-depth robust.

Example: (1,2)-reducible

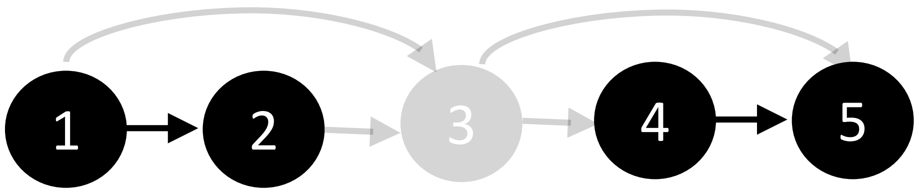


Depth Robustness

Definition: A DAG G=(V,E) is (e,d)-reducible if there exists $S \subseteq V$ s.t. $|S| \leq e$ and depth(G-S) $\leq$ d.

Otherwise, we say that G is (e,d)-depth robust.

Example: (1,2)-reducible



Attacking (e,d)-reducible DAGs

- Input: $|S| \leq e$ such that depth(G-S) = d, g > d

- **Light Phase (g rounds)**: Discard most pebbles!
  - Goal: Pebble the next g nodes in g (sequential) steps
  - Low Memory (only keep pebbles on S and on parents of new nodes)
  - Lasts a ``long” time

- **Balloon Phase (d rounds):** Greedily Recover Missing Pebbles
  - Goal: Recover needed pebbles for upcoming light phase
  - Expensive, but quick (at most d steps in parallel).

Attacking (e,d)-reducible DAGs

Algorithm 1: GenPeb (G, S, g, d)

Arguments: $G = (V, E)$, $S \subseteq V$, $g \in [\mathsf{depth}(G - S), n]$, $d \geq \mathsf{depth}(G - S)$

for $i = 1$ *to* n **do**
- Pebble node i.
- $l \leftarrow \lfloor i/g \rfloor * g + d + 1$
- **if** $i \mod g \in [d]$ **then** `// Balloon Phase`
  - $d' \leftarrow d - (i \mod g) + 1$
  - $N \leftarrow \mathsf{need}(l, l + g, d')$
  - Pebble every $v \in N$ which has all parents pebbled.
  - Remove pebble from any $v \notin K$ where $K \leftarrow S \cup \mathsf{keep}(i, i + g) \cup \{n\}$.
- **else** `// Light Phase`
  - $K \leftarrow S \cup \mathsf{parents}(i, i + g) \cup \{n\}$
  - Remove pebbles from all $v \notin K$.
- **end**

end

Main Theorem

Theorem (Depth-Robustness is a necessary condition): If G is (e,d)-reducible then is an (efficient) attack A such that

$$\mathrm{E}_{\mathrm{R}}(A) \le en + \delta gn + \frac{n}{g}nd + n\mathrm{R} + \frac{n}{g}n\mathrm{R}.$$

Main Theorem

Theorem (Depth-Robustness is a necessary condition): If G is (e,d)-reducible then is an (efficient) attack A such that

$$\mathrm{E}_{\mathrm{R}}(A) \leq en + \delta gn + \frac{n}{g}nd + n\mathrm{R} + \frac{n}{g}n\mathrm{R}.$$

Upper bounds pebbles on nodes x $\in S$, where
$|S| = e$
depth(G-S) $\leq d$

#pebbling rounds

Main Theorem

Theorem (Depth-Robustness is a necessary condition): If G is (e,d)-reducible then is an (efficient) attack A such that

$$\mathrm{E_R}(A) \leq en + \delta gn + \frac{n}{g} nd + n\mathrm{R} + \frac{n}{g} n\mathrm{R}.$$

Maintain pebbles on parents of next g nodes to be pebbled.

#pebbling rounds

Main Theorem

Theorem (Depth-Robustness is a necessary condition): If G is not (e,d)-node robust then is an (efficient) attack A such that

$$\mathrm{E}_{\mathrm{R}}(A) \leq en + \delta gn + \frac{n}{g}nd + n\mathrm{R} + \frac{n}{g}n\mathrm{R}.$$

Length of a balloon phase

#balloon phases

Max #pebbles on G
In each round of balloon phase

Main Theorem

Theorem (Depth-Robustness is a necessary condition): If G is not (e,d)-node robust then is an (efficient) attack A such that

$$\mathrm{E_R}(A) \leq en + \delta gn + \frac{n}{g} nd + n\mathrm{R} + \frac{n}{g} n\mathrm{R}.$$

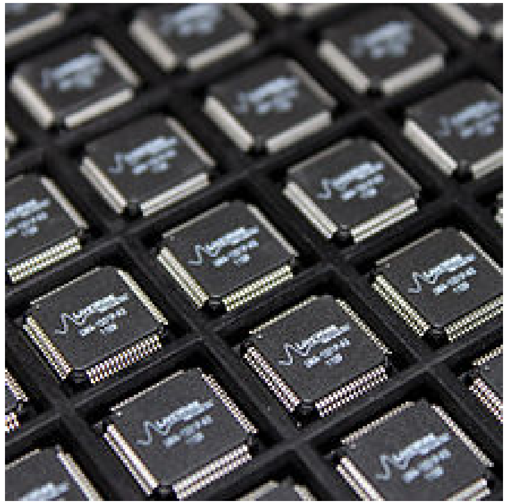
Set $g = \sqrt{nd}$

$$\mathrm{E_R}(A) = \mathrm{O}\left(en + \sqrt{n^3 d}\right).$$

In particular, $\mathrm{E_R}(A) = \mathrm{o}(n^2)$ for e,d=o(n).

Question

Are existing iMHF candidates based on depth-robust DAGs?



iMHF Candidates

- Catena [FLW15]
  - Special Recognition at Password Hashing Competition
  - Two Variants: Dragonfly and Double-Butterfly
  - Security proofs in sequential space-time model

- Balloon Hashing [CBS16]
  - Newer proposal (three variants in original proposal)

- Argon2 [BDK15]
  - Winner of the Password Hashing Competition
  - Argon2i (data-independent mode) is recommended for Password Hashing

- This Talk: Focus on Argon2i-A (version from Password Hashing Competition)
  - Attack ideas do extend to Argon2i-B (latest version)

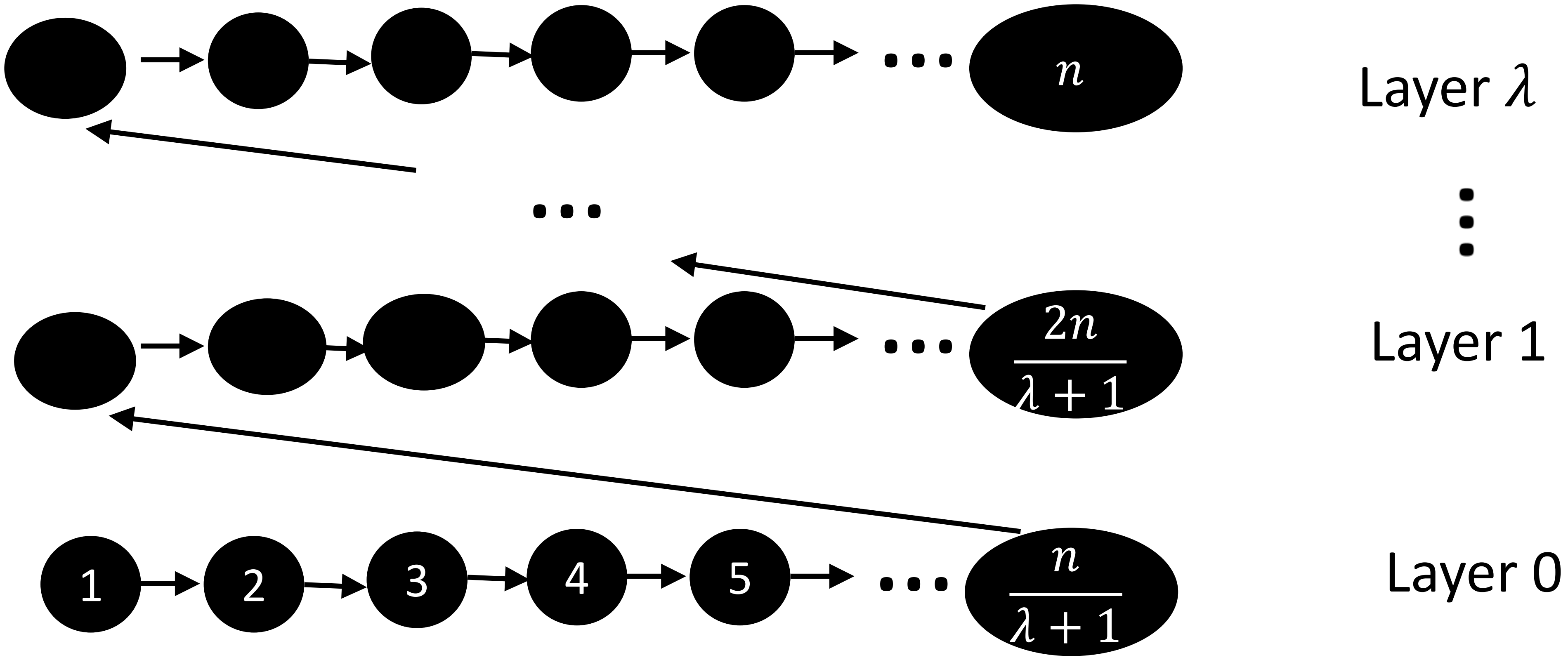
Attack Outline

- Show that any “layered DAG” is reducible
  - Note: Catena DAGs are layered DAGs
- Show that an Argon2i DAG is *almost* a “layered DAG.”
  - Turn Argon2i into layered DAG by deleting a few nodes
  - Hence, an Argon2i DAG is also reducible.

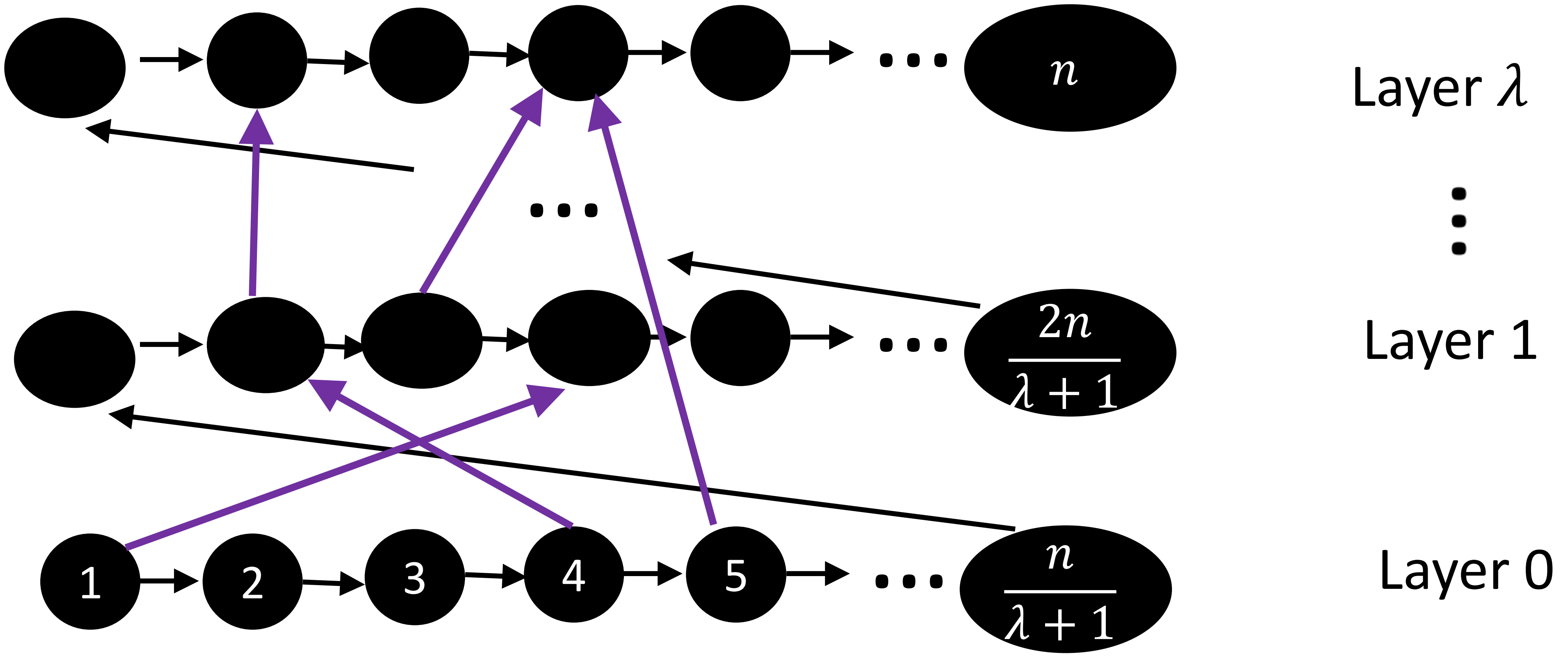
Catena

- Catena Bit Reversal DAG (BRG_λ^n)
  - λ-layers of nodes ($\lambda \leq 5$)
  - Edges between layers correspond to the bit-reversal operation
  - **Theorem[LT82]:** sST(BRG_1^n)$= \Omega(n^2)$
- Catena Butterfly (DBG_λ^n)
  - $\lambda = O(\log n)$-layers of nodes
  - Edges between layers correspond to FFT
  - DBG_λ^n is a “super-concentrator.”
  - **Theorem[LT82] =>** sST(BRG_1^n)$= \Omega\left(\frac{n^2}{\log(n)}\right)$

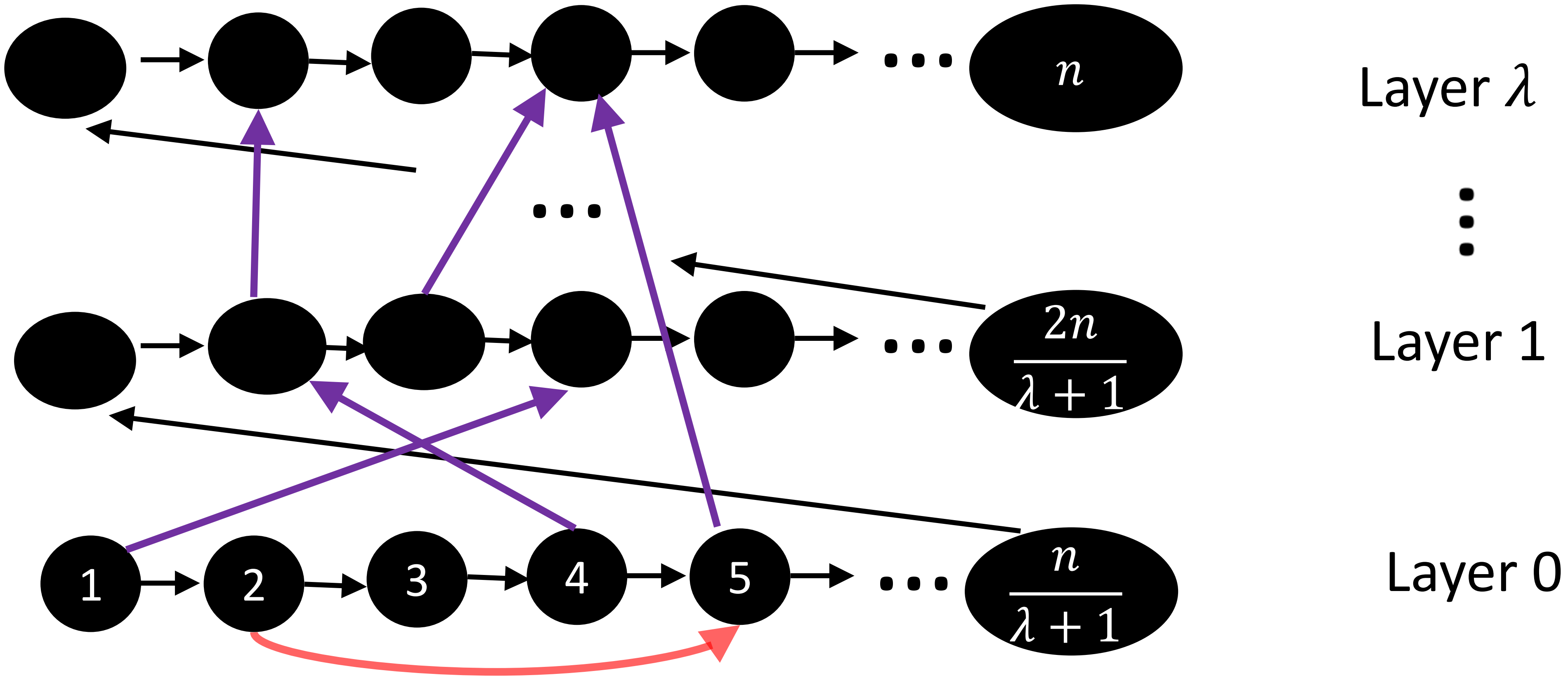
λ-Layered DAG (Catena)



λ-Layered DAG (Catena)



λ-Layered DAG (Catena)

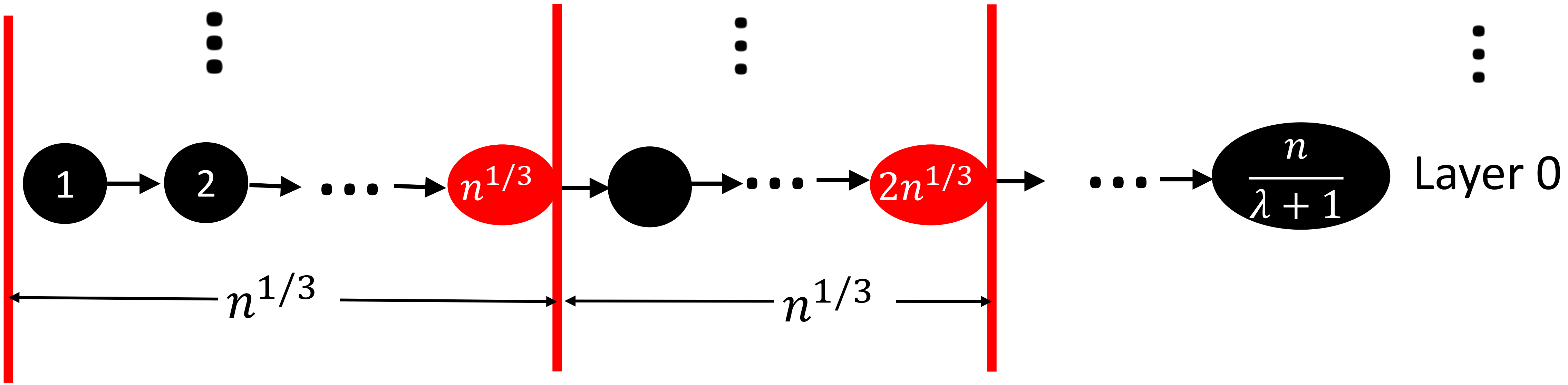


Disallowed! All edges must go to a higher layer (except for (i,i+1))

Layered Graphs are Reducible

Theorem (Layered Graphs Not Depth Robust): Let G be a λ-Layered DAG then G is $\left(n^{2/3}, n^{1/3}(\lambda+1)\right)$-reducible.

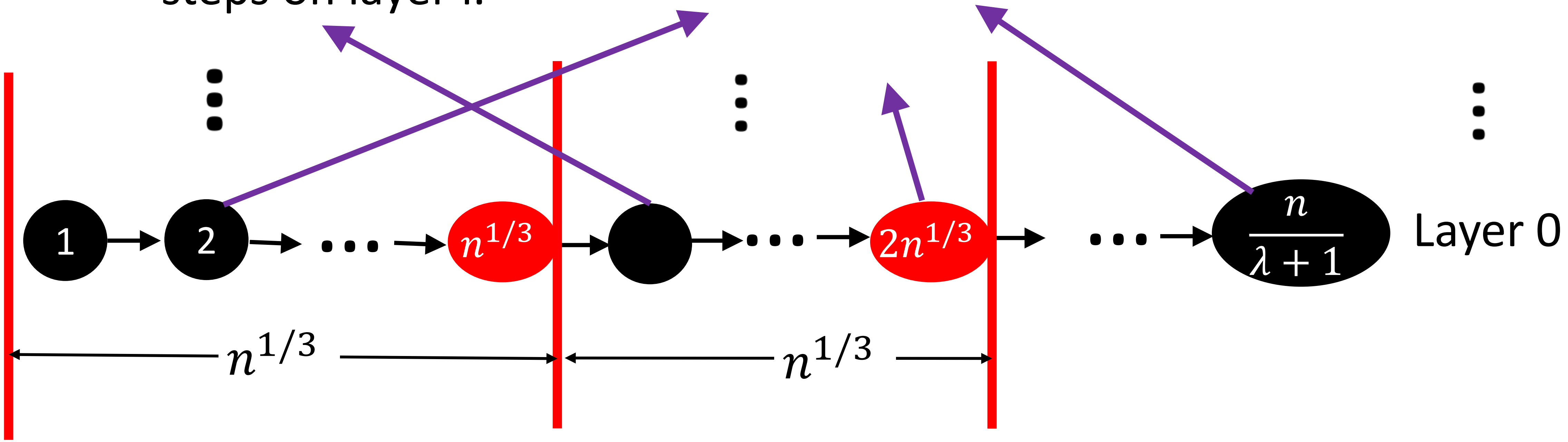
Proof: Let $\mathbf{S} = \left\{\boldsymbol{i} \times \boldsymbol{n}^{1/3} \middle| \boldsymbol{i} \leq \boldsymbol{n}^{2/3}\right\}$ any path p can spend at most $n^{1/3}$ steps on layer i.



Layered Graphs are Reducible

Theorem (Layered Graphs Not Depth Robust): Let G be a λ-Layered DAG then G is $\left(n^{2/3}, n^{1/3}(\lambda+1)\right)$-reducible.

Proof: Let $\boldsymbol{S} = \left\{\boldsymbol{i} \times \boldsymbol{n}^{1/3} \middle| \boldsymbol{i} \le \boldsymbol{n}^{2/3}\right\}$ any path p can spend at most $n^{1/3}$ steps on layer i.



Layered Graphs are Reducible

Theorem (Layered Graphs Not Depth Robust): Let G be a λ-Layered DAG then G is $\left(n^{2/3}, n^{1/3}(\lambda+1)\right)$-reducible.

Corollary: $\mathrm{E}_{\mathrm{R}}(G) \leq O\left(\lambda n^{5/3}\right)$.

Attack Quality: $\mathrm{Quality}_{\mathrm{R}}(A) = \Omega\left(\frac{n^{1/3}}{\lambda}\right)$.

Previous Attacks on Catena

- [AS15] $\mathrm{CC}(\mathrm{BRG}_1^n) \leq O(n^{1.5})$
  - Gap between cumulative cost $O(n^{1.5})$ and sequential space-time cost $\Omega(n^2)$

- [BK15] $\mathrm{ST}(\mathrm{BRG}_\lambda^n) \leq O(n^{1.8})$ for $\lambda > 1$.
- Our result $\mathrm{CC}(\mathrm{BRG}_\lambda^n) \leq O(n^{1.67})$ *

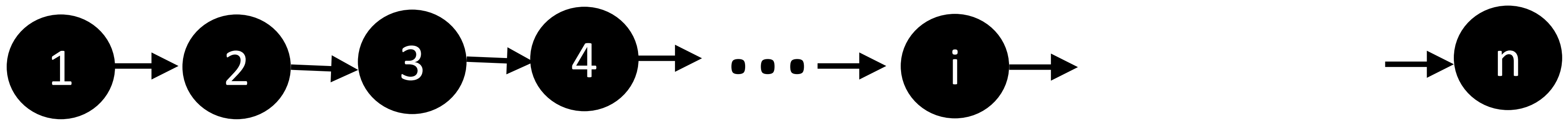
  * Applies to all Catena variants.

Argon2i [BDK]

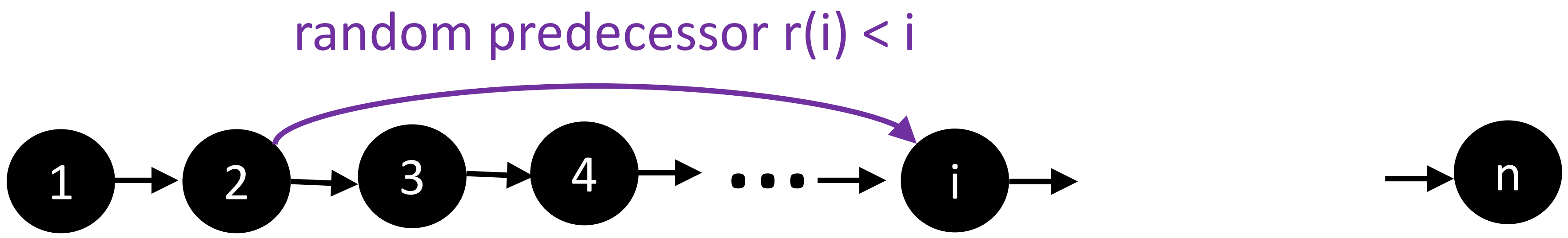
- Argon2: Winner of the password hashing competition[2015]

- Authors recommend Argon2i variant (data-independent) for password hashing.

Argon2i

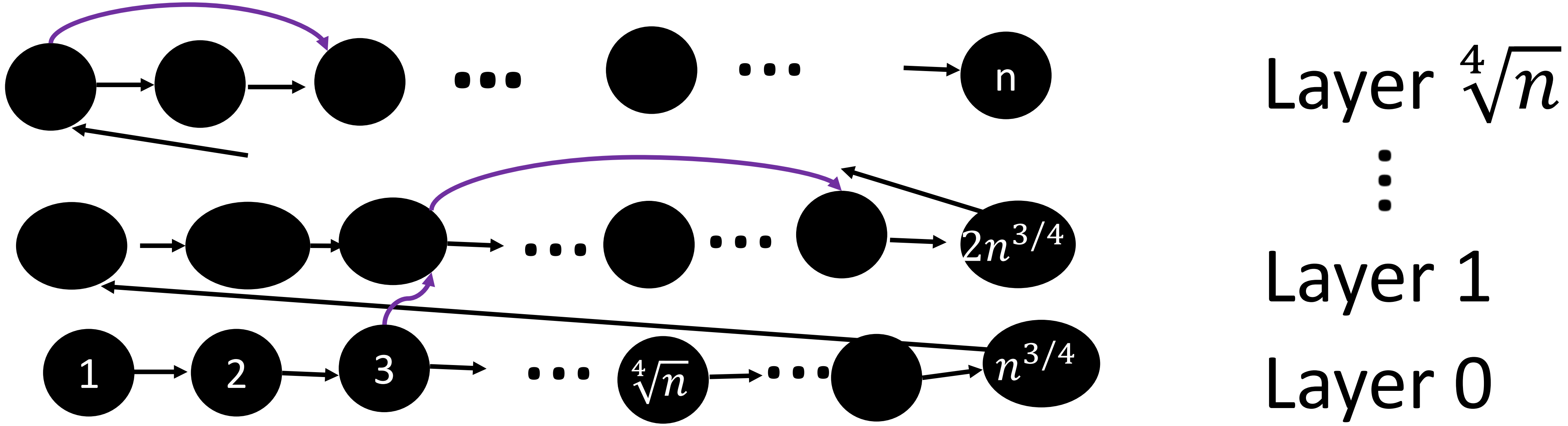


Argon2i



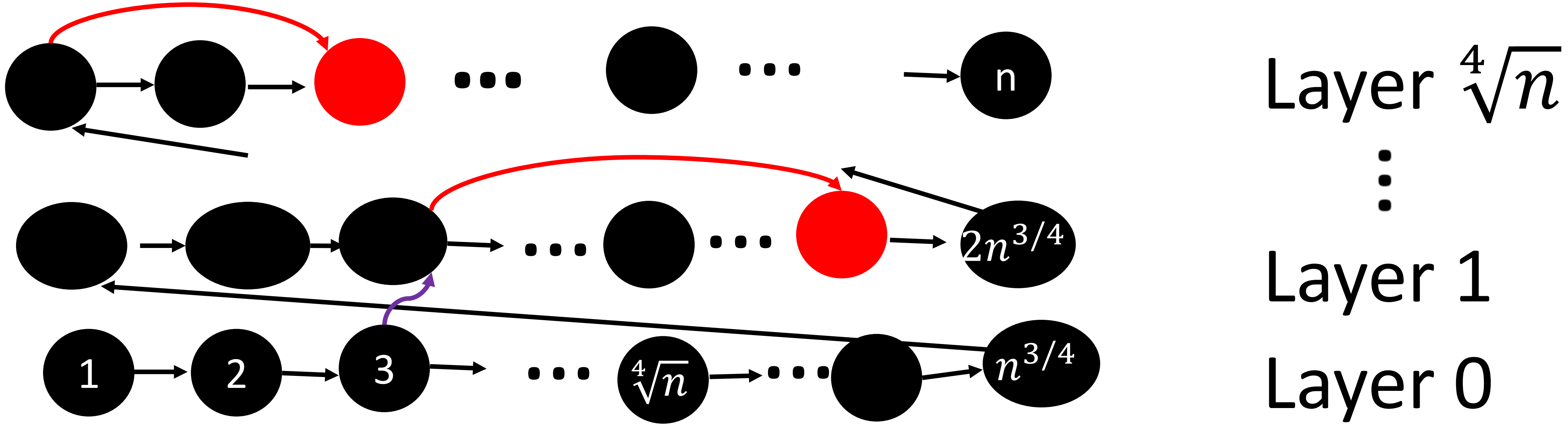
Indegree: $\delta = 2$

Argon2i is a layered DAG (almost)



Argon2i is a layered DAG (almost)

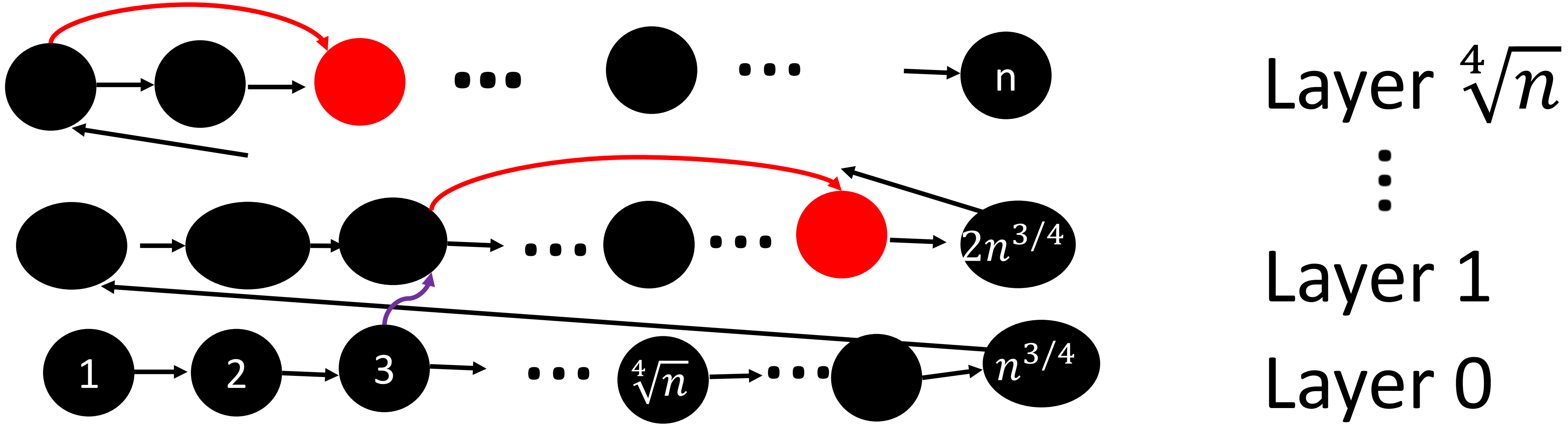
Definition: $S_2 = \{v_i \mid v_{r(i)} \text{ and } v_i \text{ in same layer}\}$



Claim: $E[S_2] = O\left(n^{3/4} \log n\right)$

Argon2i is a layered DAG (almost)

Definition: $S_2 = \{v_i |\, v_{r(i)} \text{and } v_i \text{ in same layer}\}$

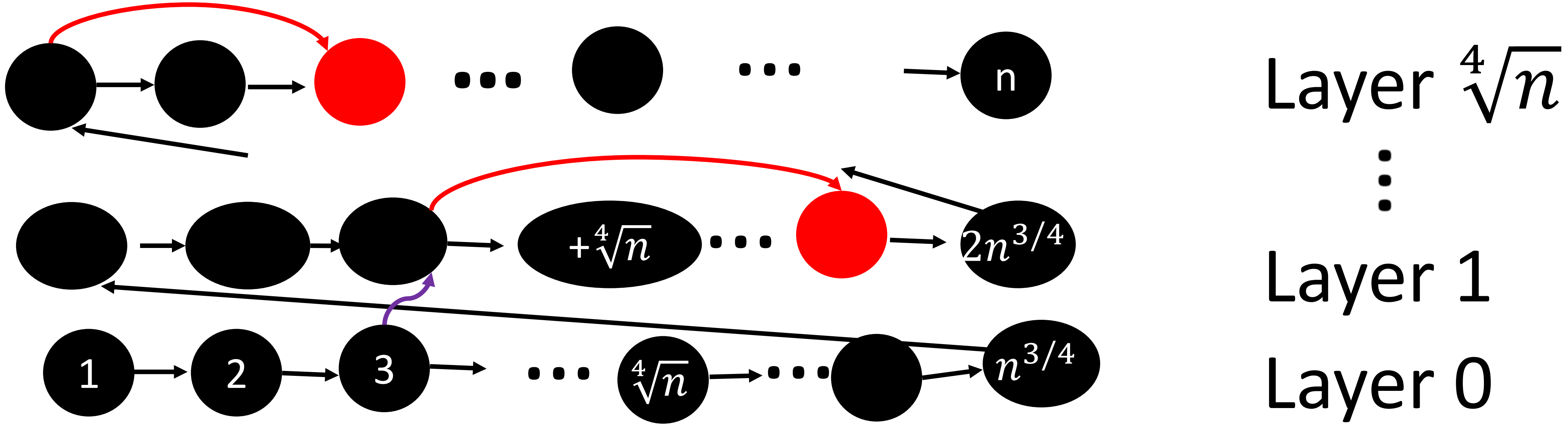


$\Pr[v \in S_2 | v \text{ in Layer } i] \le \frac{1}{i}$

$E[Layer\ i \cap S_2] \le \frac{n^{3/4}}{i}$

Argon2i is a layered DAG (almost)

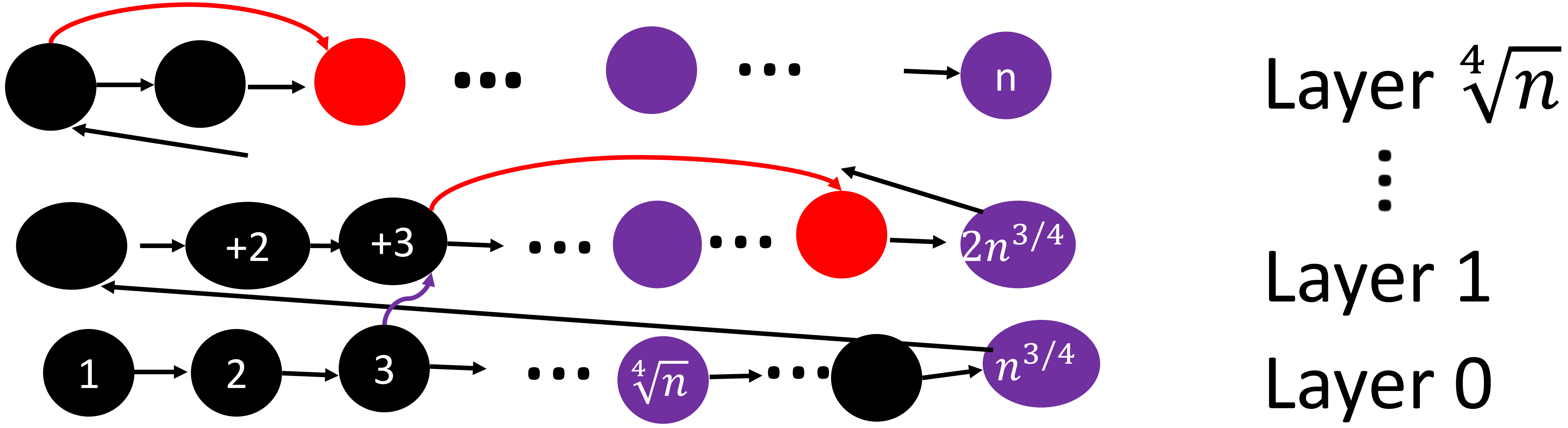
Definition: $S_2 = \{v_i \mid v_{r(i)} \text{ and } v_i \text{ in same layer}\}$



Claim: $E[S_2] = O\left(n^{3/4} \log n\right)$

Argon2i is a layered DAG (almost)

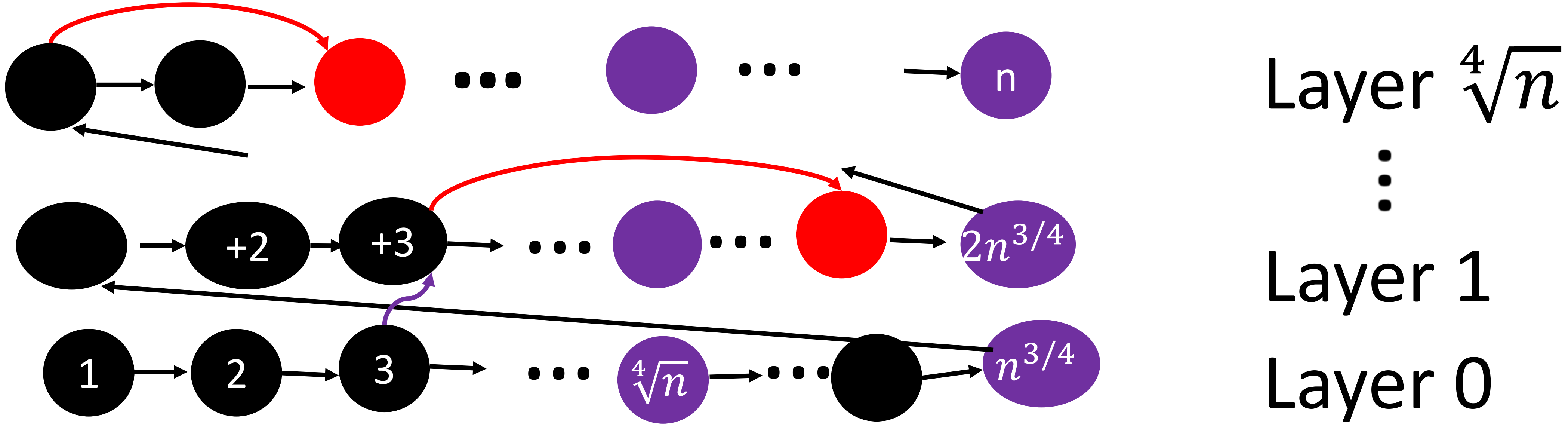
Let S = S$_1$+S$_2$



Fact: $\mathrm{E}[S] = O\left(n^{3/4} \log n\right)$ and depth(G-S)$\leq \sqrt{n}$.

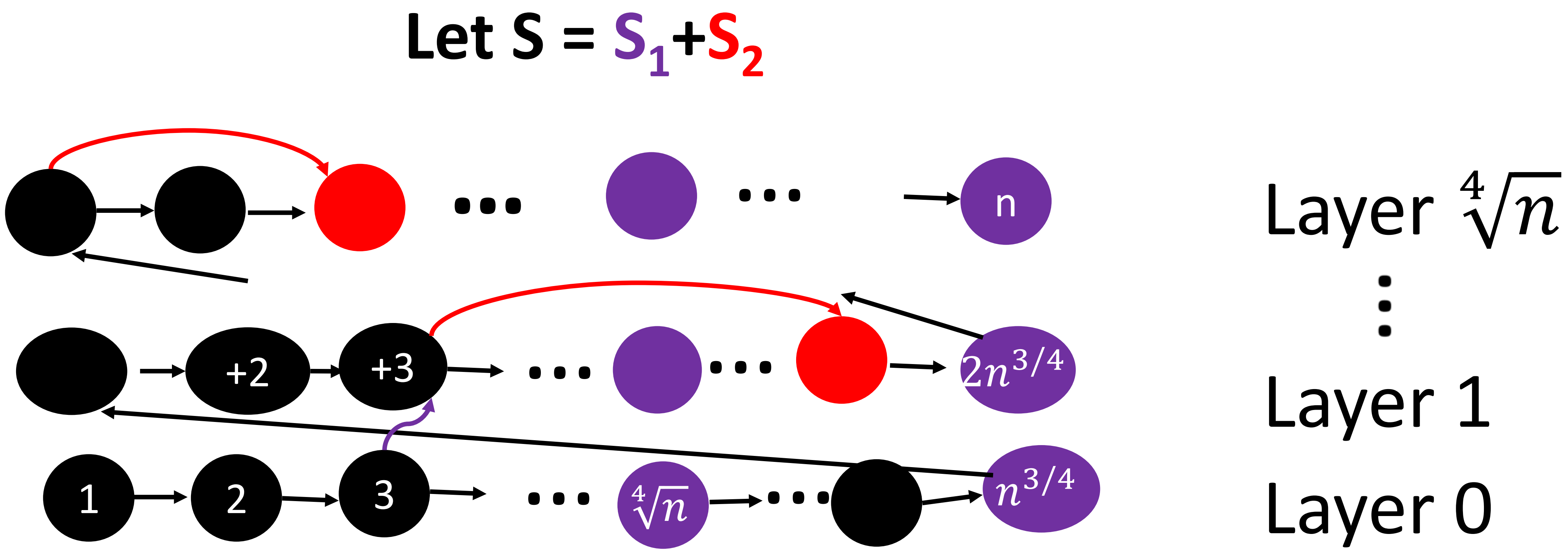
Argon2i is a layered DAG (almost)

Let S = S_1+S_2



Theorem: G is $(2n^{3/4}\log n, \sqrt{n})$-reducible with high probability.

Argon2i is a layered DAG (almost)



Corollary: $\mathrm{ER}(G) \leq O\left(n^{7/4} \log n\right)$.

$\mathrm{Quality}_{\mathrm{R}}(A) \leq \Omega\left(\frac{n^{1/4}}{\log n}\right)$.

Ideal iMHFs Don't Exist

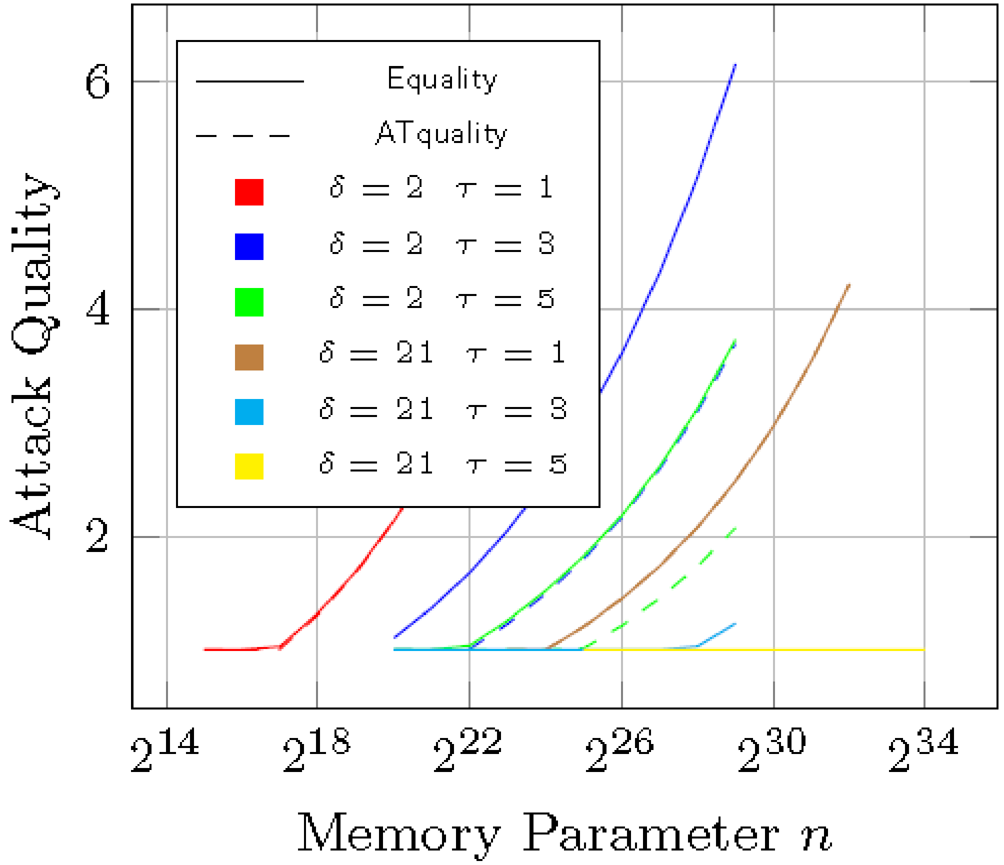
- **Thm:** If G has n nodes and constant in-degree δ=O(1) then G is :

$$\left(O\left(\frac{n \log\log n}{\log(n)}\right), \frac{n}{\log^2 n}\right)\text{-reducible.}$$

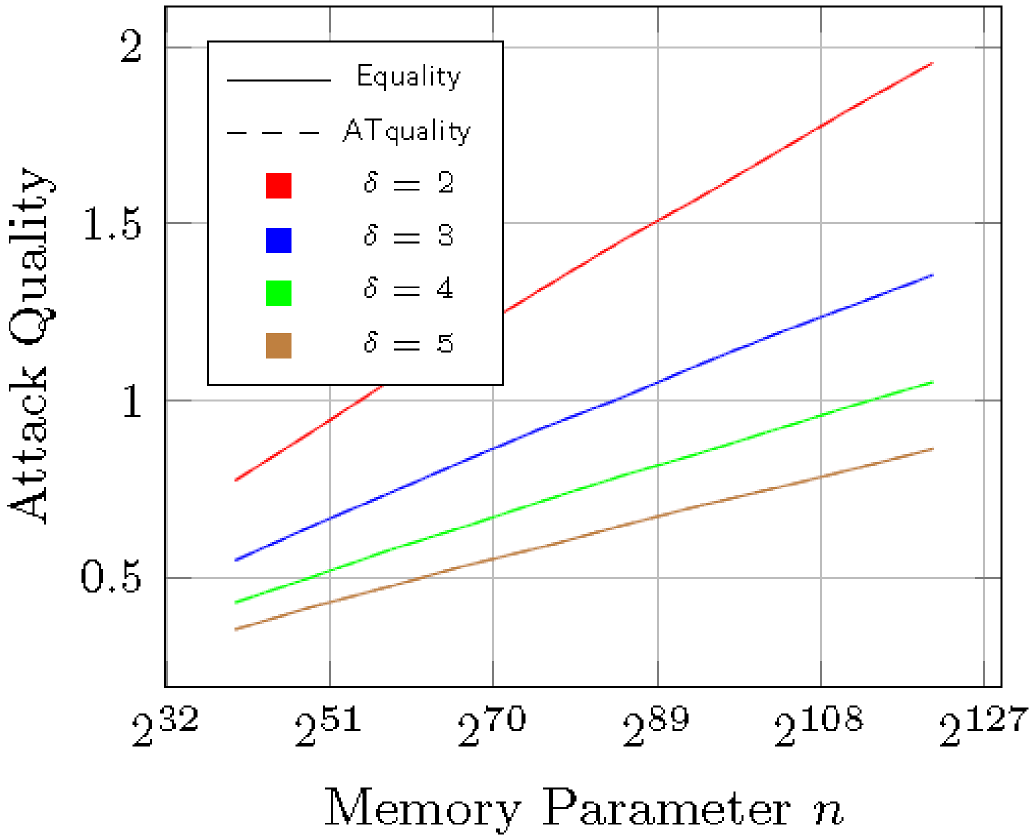
- **Thm:** If G has n nodes and constant in-degree then:

$$\forall \varepsilon > 0 \quad \mathrm{E}_{\mathrm{R}}(G) = o\left(\frac{n^2}{\log(n)^{1-\varepsilon}} + nR\right)$$

Practical Consequences (R = 3,000)



(a) Argon2i and SB



(b) Ideal iMHF


THEORY VS PRACTICE

Drama: Are the attacks `Practical'

- Argon2i team: No, at least for realis

- Recent: Argon2i-B submitted to IRT
  Task Force) for standardization.

- New Result [AB16b]:
  - New heuristics to reduce overhead by constant factor
  - Simulate the attack on real instances

New Simulation Results : [AB16b]

Parameter setting could easily be chosen when following Argon2i-B guidelines

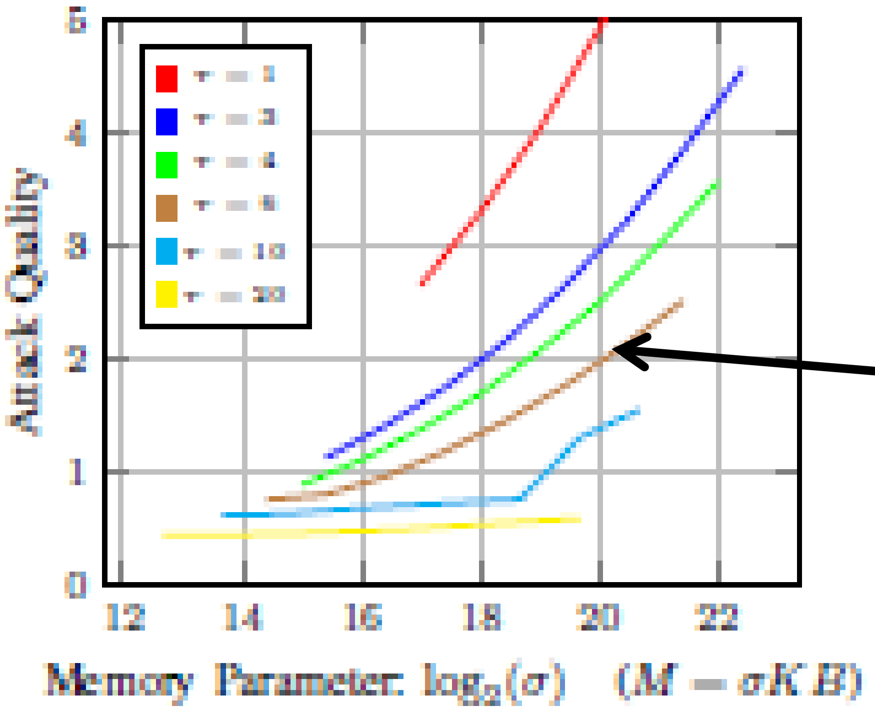


Figure 1: Argon2i-B Attack Quality

Pessimistic Argon 2i-B parameter

Attack on Argon 2i-B is practical even for pessimistic parameter ranges (brown line).

Outline

- Motivation
- Data Independent Memory Hard Functions (iMHFs)
- Attacks
- **Constructing iMHFs (New!)**
  - **Depth-Robustness is *sufficient***
- Conclusions and Open Questions

Depth-Robustness is Sufficient! [ABP16]

Key Theorem: Let G=(V,E) be (e,d)-depth robust then CC(G)$\geq ed$.

Implications: There exists a constant indegree graph G with

$$\mathrm{CC}(\mathrm{G}) \geq \Omega\left(\frac{n^2}{\log n}\right).$$

Previous Best [AS15]: $\Omega\left(\frac{n^2}{\log^{10} n}\right)$

[AB16]: For all constant indegree graphs $\mathrm{CC}(\mathrm{G}) = O\left(\frac{n^2 \log\log n}{\log n}\right)$.

Depth-Robustness is Sufficient! [ABP16]

Proof: Let $P_1, \ldots P_t$ denote an (optimal) pebbling of G. For 0< i < d define

$$S_i = P_i \cup P_{d+i} \cup P_{2d+i} \cup \cdots$$

one of the sets S_i has size at most CC(G)/d. Now we claim that

$$d \geq \text{depth}(G\text{-}S_i)$$

because any path in G-S_i must have been completely pebbled at some point. Thus, it must have been pebbled entirely during some interval of length d. Thus, G (CC(G)/d,d)-reducible. It follows that CC(G)$\geq ed$.

Proof by Picture

$$S_i = P_i \cup P_{d+i} \cup P_{2d+i} \cup \cdots$$

$$P_1, P_2, \ldots, P_{i-1}, \underbrace{P_i, P_{i+1}, \ldots, P_{i+d-1}, P_{i+d}}_{\text{d rounds}}, \underbrace{P_{i+d+1}, \ldots, P_{i+2d-1}, P_{i+2d}}_{\text{d rounds}}, \ldots.$$

Claim: $|S_i| \geq e$

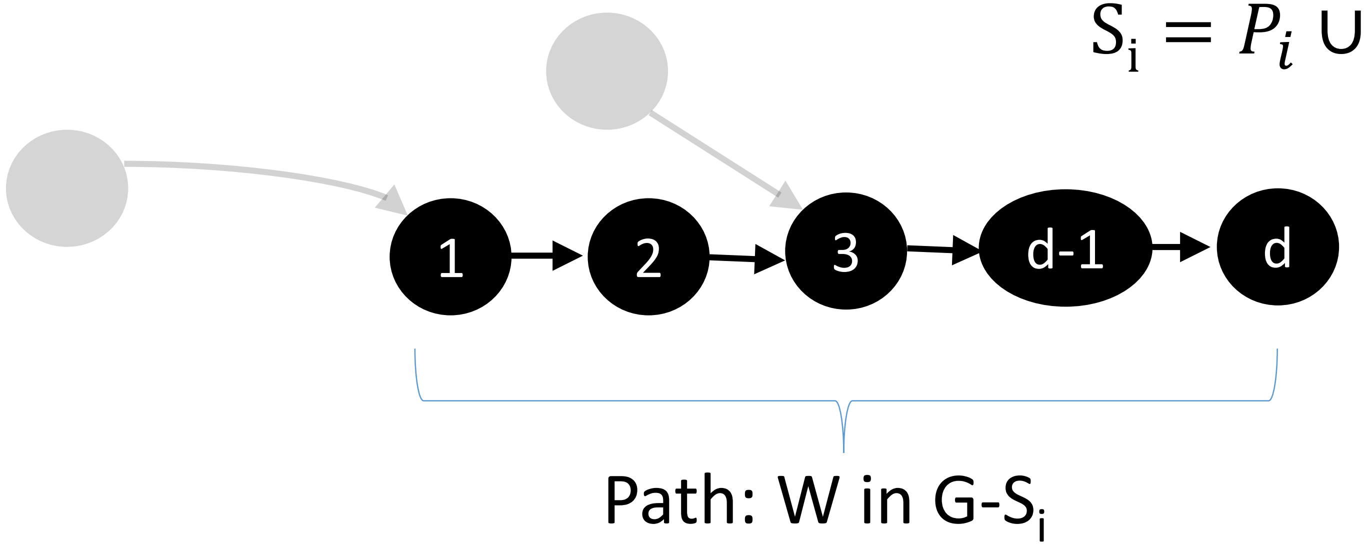
Implication

Claim: $|\mathrm{S_i}| \geq e$

$$CC(G) \geq \sum_t |P_t| \geq \sum_{i=1}^{d} |S_i| \geq \sum_{i=1}^{d} e \geq ed$$

Contradiction by Picture

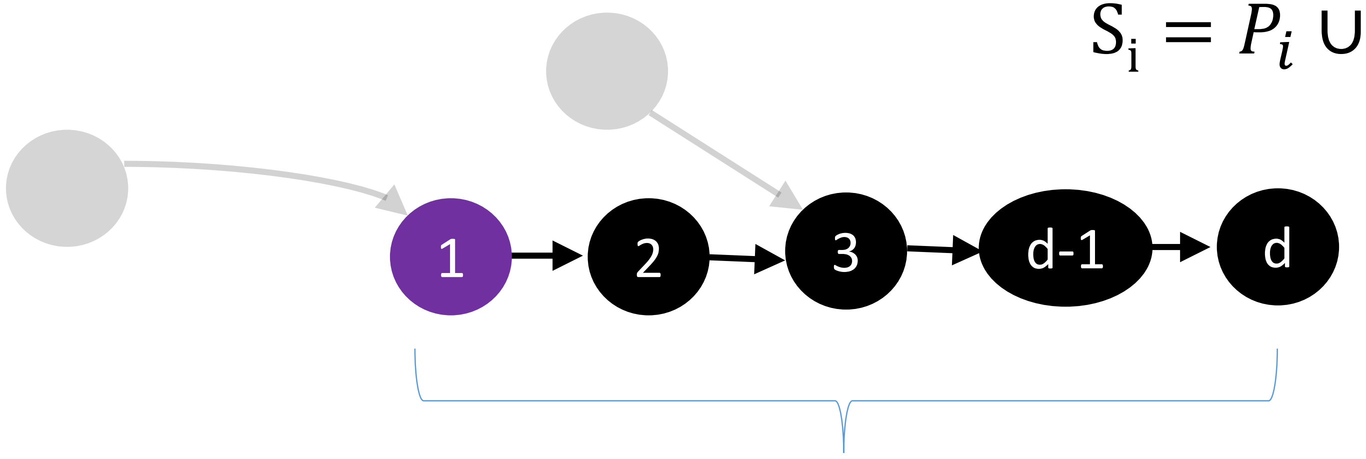
$$S_i = P_i \cup P_{d+i} \cup P_{2d+i} \cup \cdots$$



Step i: W contains no pebbles since $P_i \subset S_i$

Contradiction by Picture

$$S_i = P_i \cup P_{d+i} \cup P_{2d+i} \cup \cdots$$



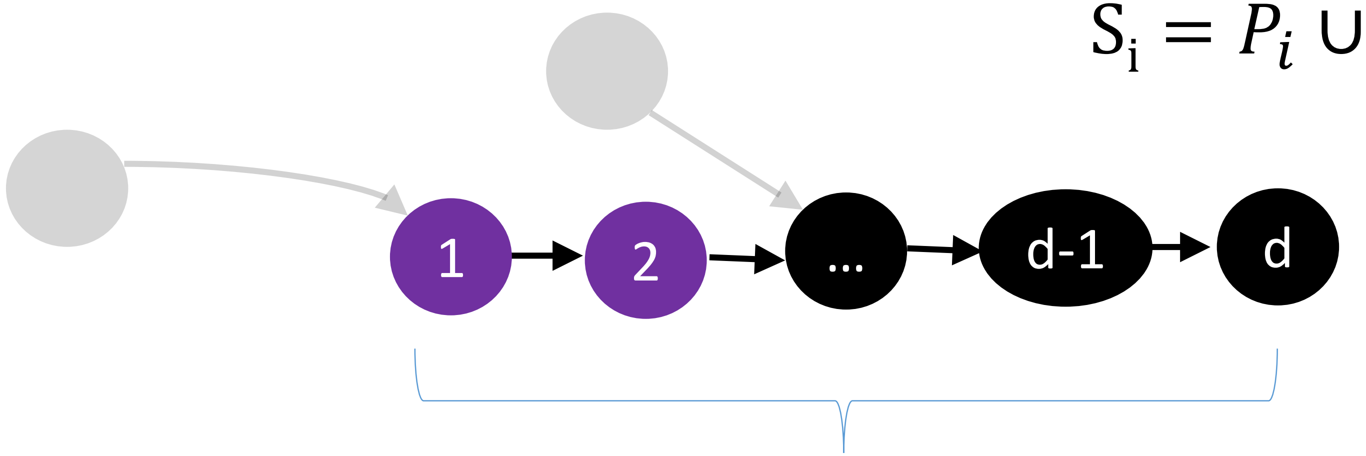
Path: W in G-S_i

Step i: W contains no pebbles since $P_i \subset S_i$

Step i+1: W-{1} contains no pebbles

Contradiction by Picture

$S_i = P_i \cup P_{d+i} \cup P_{2d+i} \cup \cdots$



Path: W in G-S_i

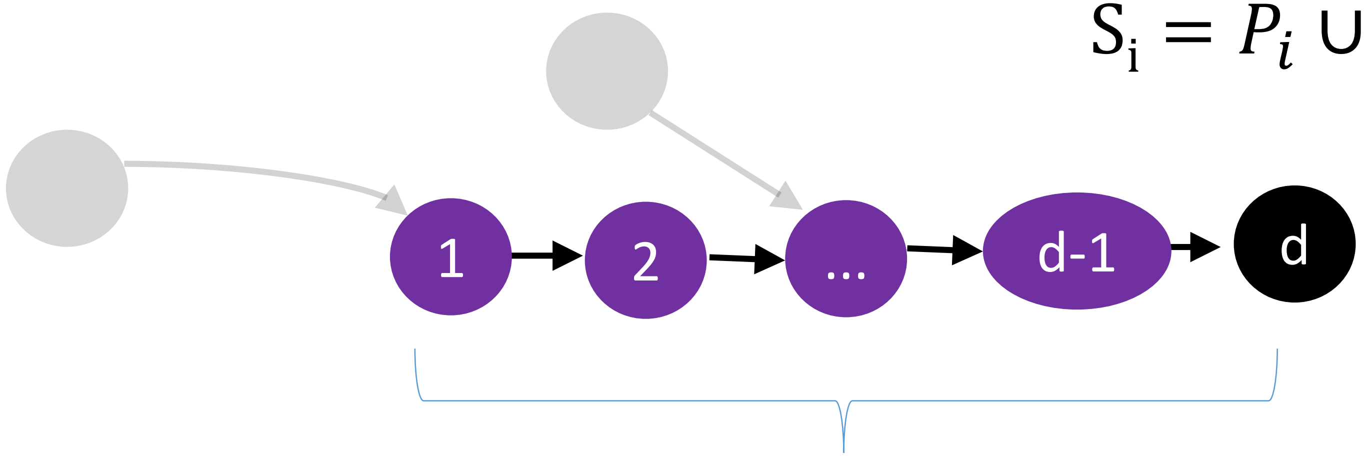
Step i: W contains no pebbles since $P_i \subset S_i$

Step i+1: W-{1} contains no pebbles

Step i+2: W-{1,2} contains no pebbles

Contradiction by Picture

$S_i = P_i \cup P_{d+i} \cup P_{2d+i} \cup \cdots$



Path: W in G-S_i

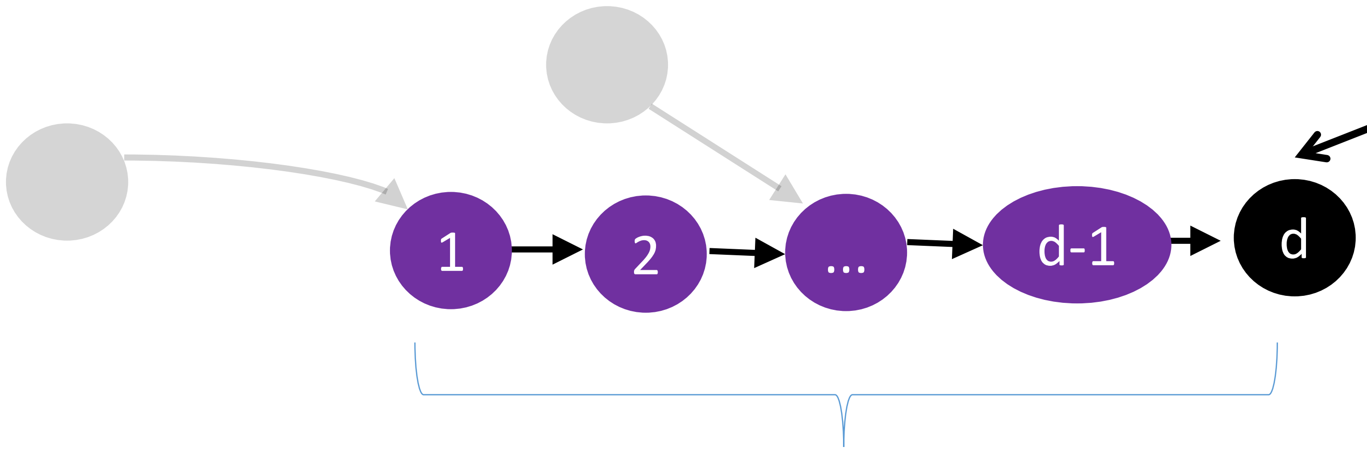
Step i: W contains no pebbles since $P_i \subset S_i$

Step i+1: W-{1} contains no pebbles

Step i+2: W-{1,2} contains no pebbles

Step i+d-1: W-{1,…,d-1} contains no pebbles

Contradiction by Picture



Path: W in G-S_i

Contradiction!
d was never pebbled.

$\therefore |S_i| \geq e$

Step i+d: W contains no pebbles since $P_{i+d} \subset S_i$

Step i+d-1: W-{1,…,d-1} contains no pebbles

Positive Result: Consequences

Theorem [ABP16]: Let G=(V,E) be (e,d)-depth robust then $\mathrm{E_R}(G) \geq ed$.

Theorem[EGS75]: There is an $\big(\Omega(n), \Omega(n)\big)$-depth robust DAG G with indegree $\delta = O(\log n)$.

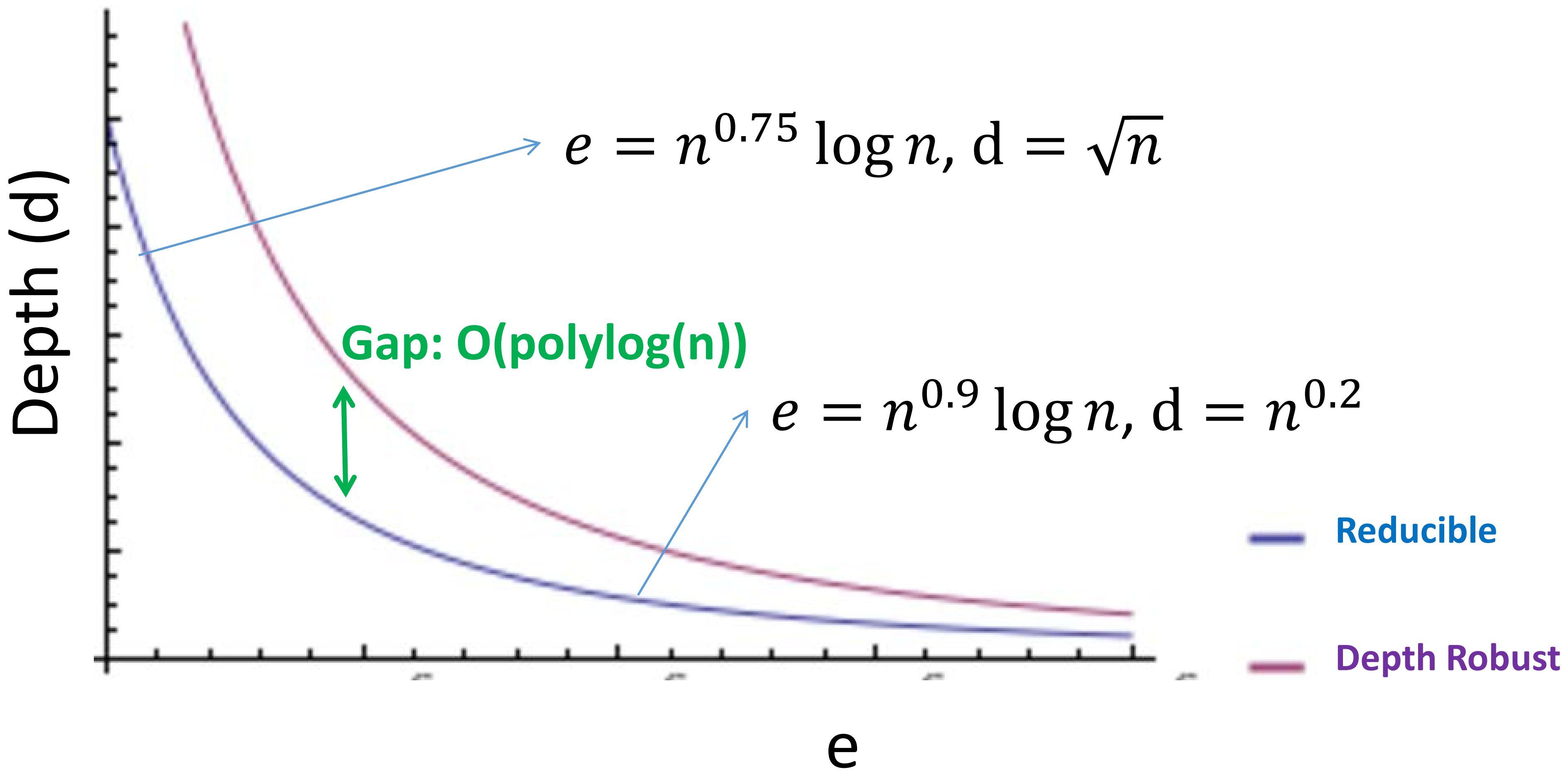
Theorem [ABP16] There is a DAG G with maximum indegree $\delta = 2$ and $\mathrm{E_R}(G) = \Omega\left(\frac{n^2}{\log n}\right)$. Furthermore, there is a sequential pebbling algorithm N with cost $\mathrm{E_R}(N) = O\left(\frac{n^2}{\log n}\right)$.

More New Results

MHF	Upper Bound	Lower Bound
Argon2i-A	$\tilde{O}(n^{1.71})$ [ABP16] $\cancel{\tilde{O}(n^{1.75})}$ [This work]	$\tilde{\Omega}(n^{1.66})$ [ABP16]
Catena	$\tilde{O}(n^{1.618})$ [ABP16] $\cancel{O(n^{1.67})}$ [This work]	$\tilde{\Omega}(n^{1.5})$ [ABP16]
SCRYPT (data dependent)	$O(n^2)$ [Naïve, P12]	$\Omega(n^2)$ [ACPRT16]

Idea: Apply our attack recursively during balloon phases

(e,d)-reducible curve for Argon2i-A



Recursive Attack

$$CC(G) \le en + \frac{n}{e} nd$$

$$CC(G) \le en + \frac{n}{e} CC(G')$$

$$CC(G) \le e_1 n + \frac{n}{e_1}\left(e_2 d_1 + \frac{n}{e_2} d_2 n\right)$$

....

Conclusions

- Depth-robustness is a necessary and sufficient for secure iMHFs
  - [AB16] [ABP16]
- Big Challenge: Improved Constructions of Depth-Robust Graphs
  - We already have constructions in theory [EGS77, PR80, …]
  - But constants matter!

More Open Questions

- Computational Complexity of Pebbling
  - NP-Hard to determine CC(G) [BZ16]
  - Hardness of Approximation?

- What is CC(Argon2i-B)?
  - Upper Bound: $O(n^{1.8})$ [AB16b]
  - Recursive attack: $O(n^{1.77})$ [BZ16b]+[ABP16]
  - Lower Bound: $\Omega(n^{1.66})$ [BZ16b]

Large Gap Remains

Thanks for Listening